Educational Services

digital

EY-8799E-SG.H001

UNDERSTANDING DECNET

STUDENT GUIDE

January 1989

CONTENTS

DECNET FUNCTIONS

DNA

CONNECTING SYSTEMS

USING THE NETWORK CONTROL PROGRAM

ROUTING

USING SET HOST, PHONE AND MAIL

CREATING NETWORK PROCESSES

REMOTE FILE ACCESS

OBJECTS

X.25 AND X.29 MODULES

DATABASES

MONITORING

FIGURES

TABLES

EXAMPLES

DECNET FUNCTIONS

DECNET FUNCTIONS

INTRODUCTION

DECnet is the collective name for the software products that allow the interconnection of various DIGITAL operating systems in a structure called a network. DECnet is the implementation of the DIGITAL Network Architecture (DNA).

This module presents an overview of the various capabilities of DECnet.

OBJECTIVES

At the end of this module the student will be able to understand the possibilities of DECnet.

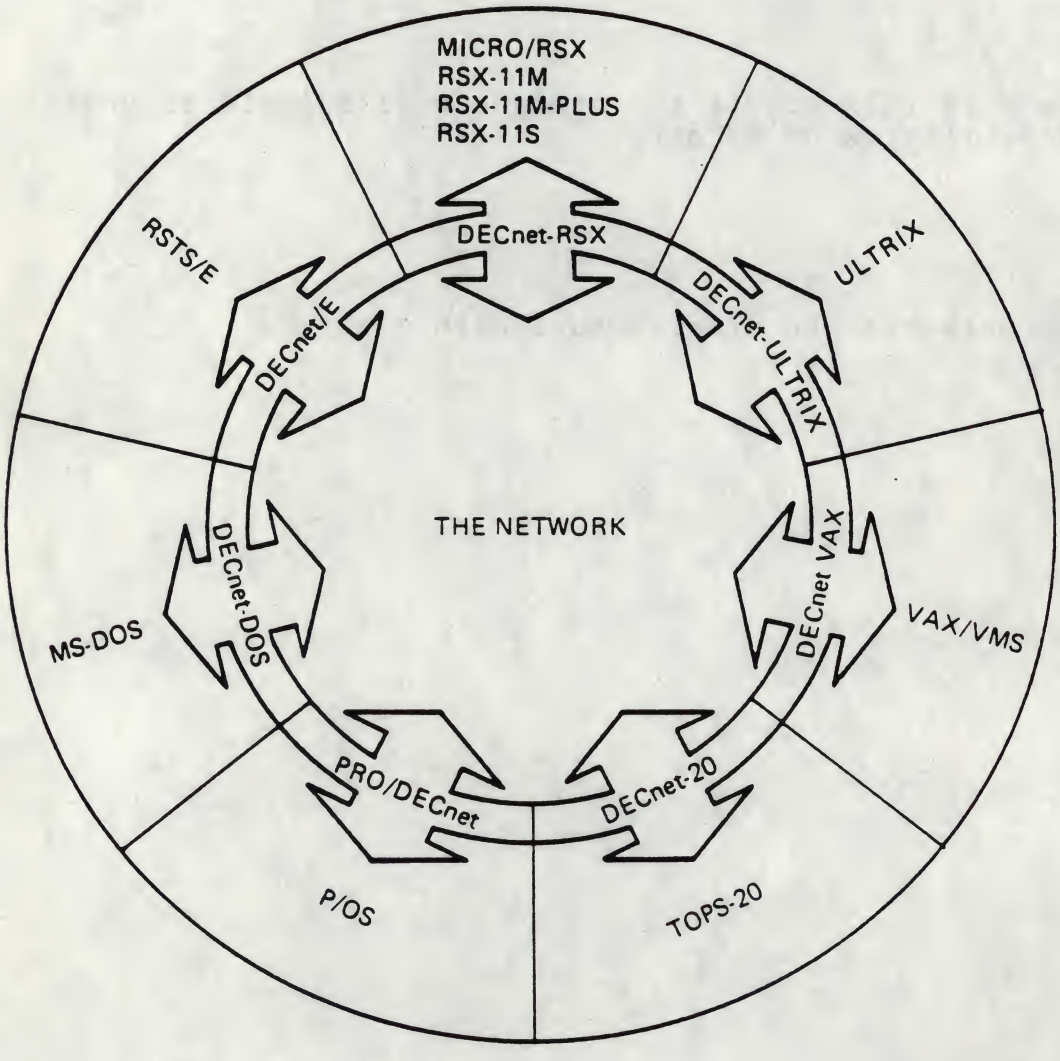
RESOURCE

Digital's Networks: An Architecture With A Future.

WHAT IS DECNET ?

DECnet is a family of software products that supports the connection of two or more DIGITAL computers to form a network. In figure 1-1, DECnet allows computer systems running many different operating systems to communicate with each other by means of a network.

Figure 1-1: DECnet Implementations



DECNET FUNCTIONS

DECnet networks may actually be considered to be one large computing system. This system provides the computer user with access to the resources available throughout the network. There are several functions and applications available to the user of a DECnet network. The following sections describe each of these network functions.

Remote File Access

DECnet allows terminal users and user-written programs to manipulate files on remote network nodes. Remote files may be accessed just as if they were located in the local node's file system.

The terminal user may issue DIGITAL Command Language (DCL) commands that perform the following remote file functions:

- Copy files to or from a remote file
- Delete remote files
- Submit and execute command files at a remote node
- Append files to local or remote files
- List remote directories
- Queue files to a remote printer

Terminal-to-Terminal Communication

DECnet expands the capability of the Phone Utility to allow users on two different nodes to send and receive data interactively.

Network Mail

The Mail Utility allows the transmission of text messages between users of a standalone VAX system. DECnet enhances the Mail capability by allowing messages to be transferred between users on different systems in the same DECnet network.

Network Virtual Terminal

The Network Virtual Terminal facility, allows a terminal user on a DECnet node to establish a connection to a remote DECnet node. Once the connection is established the local terminal acts as if it were physically connected to the remote node.

Remote Resource Sharing

DECnet allows local node users to share resources available on other nodes. Peripheral devices such as disks and printers may service several nodes in a network. This capability reduces overall hardware costs and increases peripheral utilization.

Downline Loading

DECnet provides downline loading services for memory-only remote nodes with no local disk storage. Downline loading involves transferring software from the file system device on a DECnet node to a remote system. The software transferred may be an operating system image or a task image.

Downline System Loading

The initial memory image of the operating system for a remote node may be stored on a file system device and loaded on request into an adjacent target node.

Downline Task Loading

DECnet supports downline loading task images to RSX-11S memory-only systems or terminal servers. The initial RSX-11S task image is stored on the DECnet file system device and can be loaded into an adjacent RSX-11S target node. Tasks already executing on the RSX-11S system can be checkpointed to the DECnet host file system and later returned to memory in the RSX-11S node.

Upline Memory Dumping

The upline dumping function supported by DECnet can be used to aid in troubleshooting an unattended memory-only system such as RSX-11S. When the remote node begins to fail, DECnet's upline dumping facility automatically sends a copy of the node's system image to an adjacent DECnet-VAX node. This system image can then be examined in an attempt to determine the cause of the system crash.

Task-to-Task Communication

Task-to-Task communication is a function common to most DECnet implementations. It allows two programs running under the same or different operating systems to communicate with each other regardless of the programming languages used. Using DECnet-VAX, a VAX/VMS program written in DCL, VAX MACRO, or a native mode high-level language can exchange messages with other programs on the same node or on remote nodes in the same network. DECnet imposes no special data formatting requirements on the user.

Network Management Facilities

DECnet implements the Network Control Program (NCP) to provide three primary network management functions:

- Display performance statistics and error information
- Control network components
- Test network operation

These functions can be performed at the local node or executed at remote nodes that support NCP functions.

An operator can monitor the status of DECnet activity at any node in the network. Statistics are available on the node itself as well as on the communication lines. NCP event logging allows significant network events to be reported to a terminal device or a disk file.

NCP commands are used to perform network control functions such as starting and stopping the network, enabling communication lines, changing network characteristics, and downline loading remote systems.

NCP is also used to test network components by enabling transmission and reception of test messages over individual lines, either between nodes or through other controller loopback arrangements. The messages can then be compared for possible errors.

SUMMARY

Within DECnet the user has access to the resources available throughout the network. The functions and applications to the user are:

1 o Remote File Access

2 o Terminal-to-Terminal Communication

3 o Network Mail

4 o Network Virtual Terminal

5 o Remote Resource Sharing

6 o Downline Loading

7 o Downline System Loading

8 o Downline Task Loading

9 o Upline Memory Dumping

10 o Task-to-Task Communication

11 o Network Management Facilities

DNA

INTRODUCTION

DECnet is the implementation of the DIGITAL Network Architecture (DNA).
DNA is the framework for all Digital communications products. DNA consists logically out of several layers. The several network functions which have to take place are performed by modules, residing in these layers.

In this module the functions of these layers are discussed.

OBJECTIVES

At the end of this module the student will be able to describe to dataflow through the various DNA-layers and to point out which protocols have been used.

RESOURCE

Digital's Networks: An Architecture With A Future

DIGITAL NETWORK ARCHITECTURE (DNA)

The DIGITAL Network Architecture (DNA) is the group of specifications within which DIGITAL designs its networking products. There are many implementations of DECnet, one for each of several DIGITAL operating systems. Each implementation varies slightly from the others. However, the majority of features and capabilities are common to all DECnet implementations.

Each DECnet implementation fits into the DNA architectural structure. This structure specifies that each group of similar network functions must be grouped into an entity called a layer. There are eight layers defined by DNA:

- User Layer
- Network Management Layer
- Network Applications Layer
- Session Control Layer
- End Communication Layer
- Routing Layer
- Data Link Layer
- Physical Link Layer

Each layer contains modules responsible for logically related functions required in the communication process between network nodes.

The architecture specifies two kinds of relationships between modules:

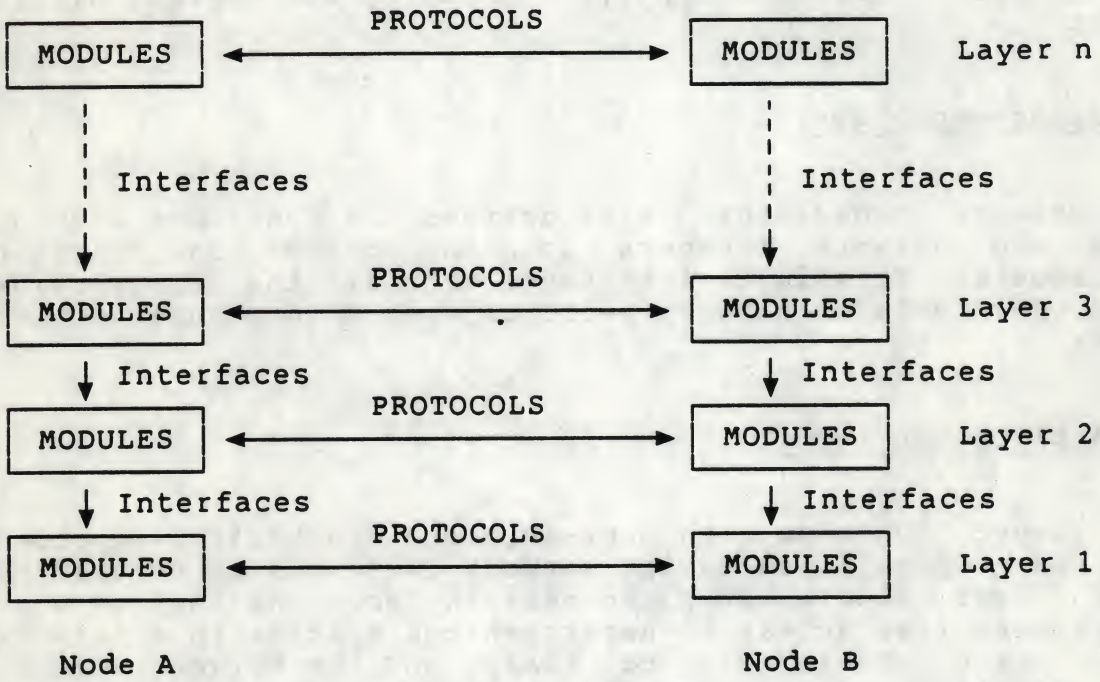
Interfaces

Interfaces are the relationships between different modules that are usually in the same node. Typically, a module in one layer interfaces with a module in the layer immediately below to receive a service and with a module in the layer immediately above to provide a service.

Protocols

Protocols are the relationships between equivalent modules that are usually in different nodes. These protocols consists of messages with specific formats and rules for exchanging the messages.

Figure 2-1: Basic DNA Structure



DECNET LAYERS

DNA defines the following layers, described in order from the highest to the lowest.

User Layer

The User layer includes all user-written programs and user level services that access the network, network services that directly support user and application tasks, and overall system management.

Network Management Layer

The Network Management layer defines the functions used by operators and network managers to plan, control, and maintain DECnet networks. This layer interfaces with all the layers below it to gather data on network performance and to change network parameters.

Network Application Layer

This layer contains both user-supplied and DIGITAL-supplied modules. Its purpose is to define network functions to be used by the User layer. DIGITAL-supplied network functions include support for remote file access to heterogeneous systems in a network using the Data Access Protocol (DAP), and the Network Virtual Terminal Facility that implements the Command Terminal Protocol (CTERM) to allow remote logins to other DECnet nodes.

Session Control Layer

The Session Control layer defines the system-dependent aspects of process-to-process communication. Its functions include name-to-address translation, process addressing, and access control.

End Communication Layer

This layer handles the system-independent aspects of communications, including connection management, data flow control, end-to-end error control, and segmentation/reassembly of user messages. This layer guarantees delivery of messages from one end of the logical link to the other.

Routing Layer

The Routing layer software defines the mechanism for routing user data from the sending node to the receiving node. It also provides network congestion control and packet lifetime control.

Data Link Layer

The Data Link layer defines the mechanism for creating an error-free communication path between adjacent nodes. This layer contains separate modules for each type of network connection and supports connections using:

- DIGITAL Data Communications Message Protocol (DDCMP)
- Ethernet
- X.25
- Computer Interconnect

Physical Link Layer

The Physical Link layer defines the physical aspects of transferring data over a transmission line. This includes defining how device drivers and communications hardware should be implemented. Functions of this layer include monitoring channel signals, clocking on the channel, handling hardware interrupts, and informing the Data Link layer when a transmission has been completed.

DECNET PROTOCOLS

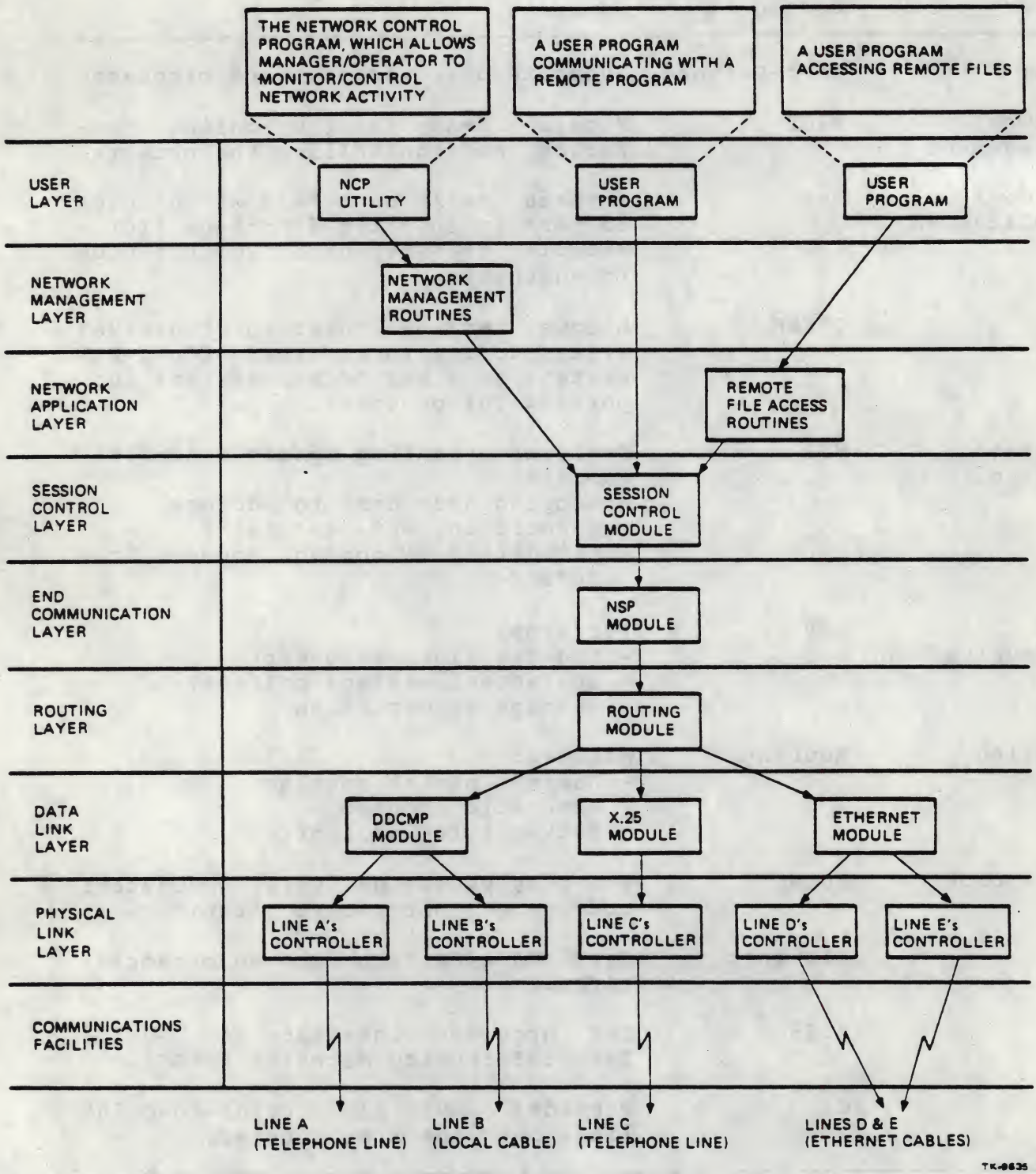
A protocol is a set of rules that defines the communication between equivalent layers residing in different network nodes. For example, modules in the Network Application layer on one node communicate with modules in the Network Application layer on a different node using the Data Access Protocol (DAP). Some DNA layers support more than one function and, therefore, include multiple protocols. An example is the Data Link layer that supports the DDCMP, X.25, Ethernet, and CI protocols.

Table 2-1 summarizes the DNA software layers (excluding the Physical Link layer), protocols used between layers, and the functions of each layer.

Table 2-1: Phase IV DNA Layers

Layer	Major Protocols	Comments
User	User-Defined	Supports user services and programs.
Network Management	NICE	Provides means for configuring, monitoring and controlling the network.
Network Application	DAP	Invokes tasks on behalf of the user to perform specific functions (for example, transfer data from one node to another).
	CTERM	Allows terminal user on DECnet-VAX system to log in to other DECnet-VAX systems or other DECnet systems supporting the protocol.
Session Control	NSP	Performs operating system dependent functions: - Mapping node name to address - Identifying end-user tasks - Validating incoming connect requests
End Communication	NSP	Performs: - Logical link management - Guaranteed message delivery - Message segmentation
Routing	Routing	Handles: - Message packet routing - Congestion control - Packet lifetime control
Data Link	DDCMP	Provides packet delivery to adjacent node on a nonbroadcast circuit.
	Ethernet	Uses CSMA/CD technique on broadcast circuit.
	X.25	DLM provides interface to Public Packet-Switching Networks (PPSN).
	CI	Provides multiple point-to-point links over the Star coupler.

Figure 2-2: DNA Modules Resident in a Typical DECnet Node



DATA FLOW

In a DECnet network, information flows from a user on one node to a user or a program on a different node. The process that occurs is similar to mailing a letter. The user message is the letter and each DECnet layer adds information to the message forming an "envelope". This added information is used at the receiving node to process the message and deliver it to the proper user.

Figure 2-3: Data Developing

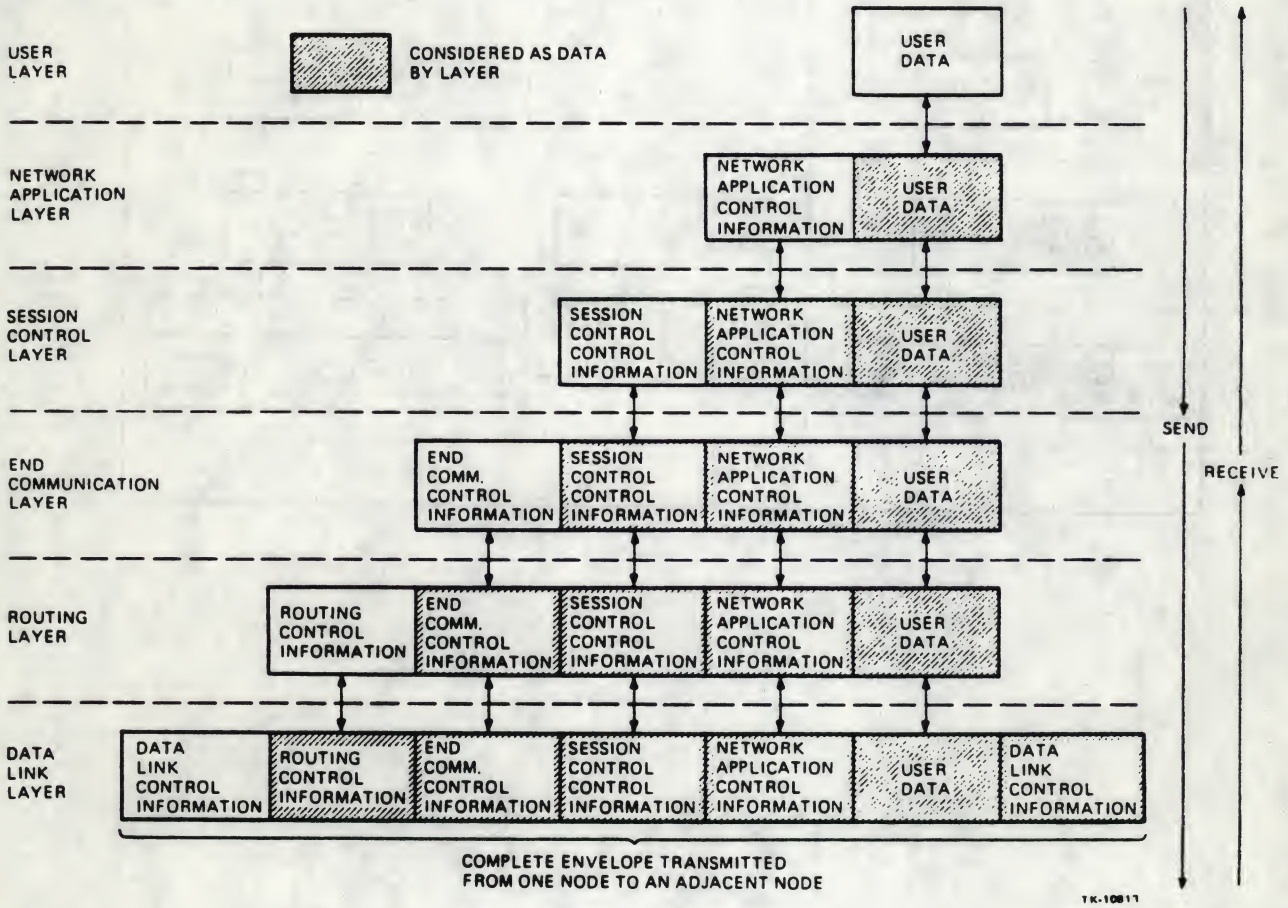
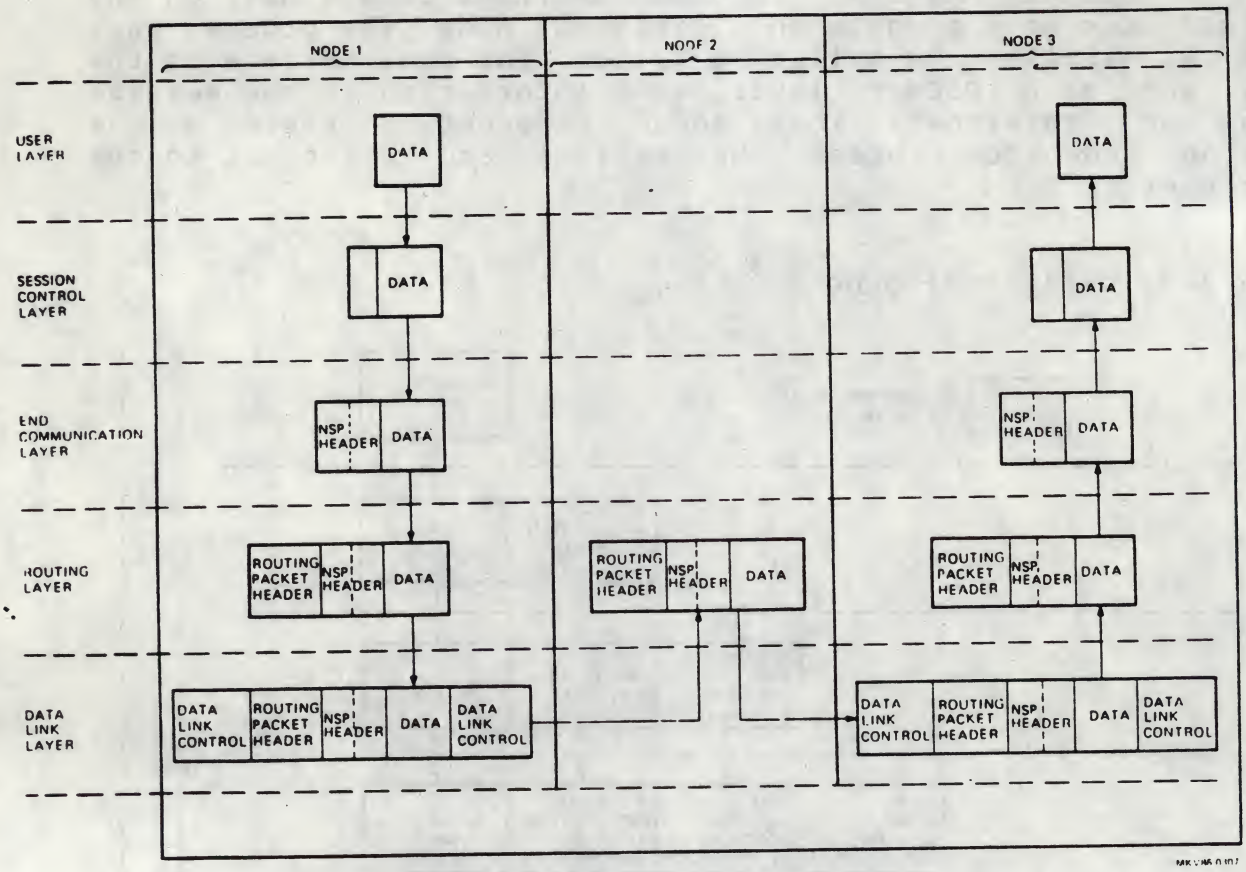


Figure 2-4: Data Flow Between Nonadjacent DECnet Nodes



DECNET PHASES

Digital has been developing networking products based on DNA since 1975. This development effort has occurred in a series of phases. So far, there have been four phases. The products in each new phase are fully compatible with those produced in the previous phase.

Phase III

Functions that DECnet Phase III systems can perform include:

- Task-to-task communication
- Remote file access
- Terminal-to-terminal communication
- Network management
- Downline loading
- Upline dumping
- Loopback testing

The following DECnet products provide Phase III capabilities:

- DECnet-10
- DECnet/E
- DECnet-RT

Phase IV

In addition to being backward-compatible with Phase III, Phase IV features include:

- Ethernet-support
- Communications server products
- Large networks
- Gateway-support to X.25 nodes
- Gateway-support to SNA-nodes

The following DECnet products provide Phase IV capabilities:

- DECnet-VAX
- DECnet-RSX
- DECnet-20
- PRO/DECnet

SUMMARY

The DIGITAL network Architecture is the group of specifications within which DIGITAL designs its products. DNA describes several functions grouped into layers.

The layers defined in DNA are:

- o User Layer
- o Network Management Layer
- o Network Applications layer
- o Session Control Layer
- o End Communication Layer
- o Routing Layer
- o Data Link Layer
- o Physical Link Layer

So far, there have been four phases of DECnet. At this moment the last phase is Phase IV. One of the main new capabilities of Phase IV is the support for Ethernet.

CONNECTING SYSTEMS

CONNECTING SYSTEMS

INTRODUCTION

There are several ways to connect some computer systems. Both hardware and software are needed. The hardware: an interface, and some pieces of software: the implementation of the protocols we are using.

This module deals with the different ways to connect two systems. When these two systems do not use the same protocols a gateway is needed. We can also connect a terminal server to a host node. Then a different protocol is used, the LAT protocol. To specify more effectively the way how the systems are connected together we use the terms LINE and CIRCUIT.

OBJECTIVES

At the end of this module the student will understand:

- The different ways to connect two systems
- The concepts of Ethernet
- The relation Ethernet - LAT
- The concepts of CIRCUITS and LINES

RESOURCES

Digital's Networks: An Architecture With A Future
VAX/VMS Networking Manual

NETWORK CONNECTIONS

DECnet provides the ability to configure nodes in several ways to form a network. Depending on the connection configuration, the network may be a local area network (LAN), a wide area network (WAN), or a combination of the two. There are several ways individual nodes can be connected to other nodes in the network.

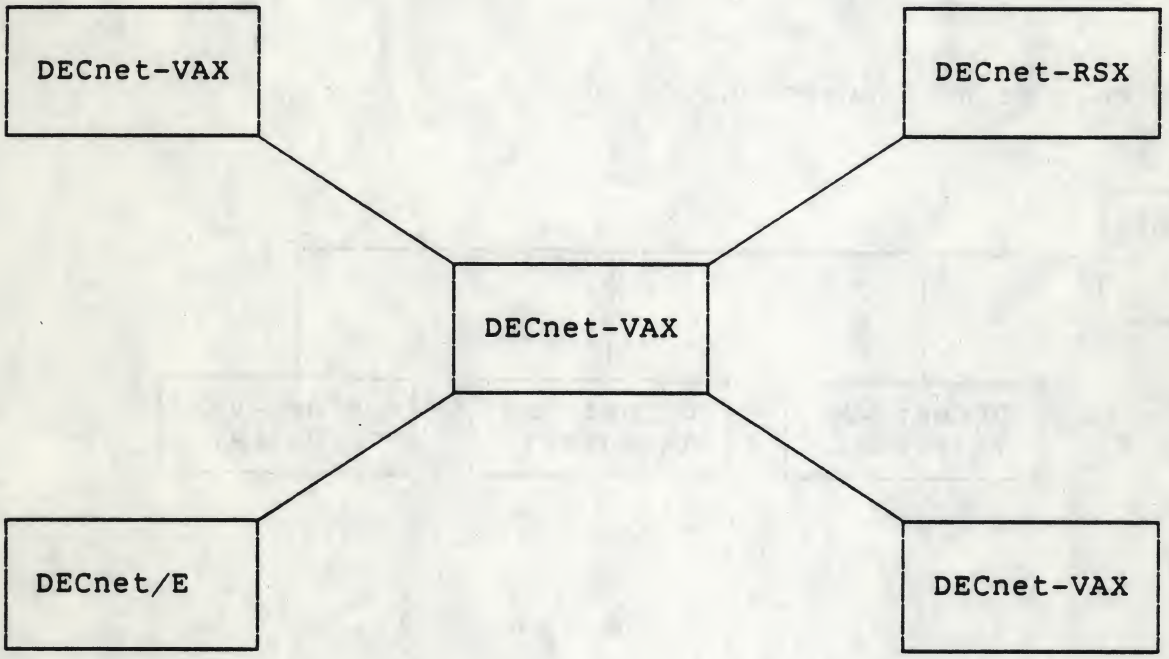
In DECnet are supported:

- Point-to-point
- Multipoint
- Ethernet
- CI

Point-to-Point

A point-to-point connection is a direct connection between two network nodes where nodes always have access to the communication line linking them. Nodes can be linked together using point-to-point connections to form many types of network configurations.

Figure 3-1: Network of Point-to-Point Connections

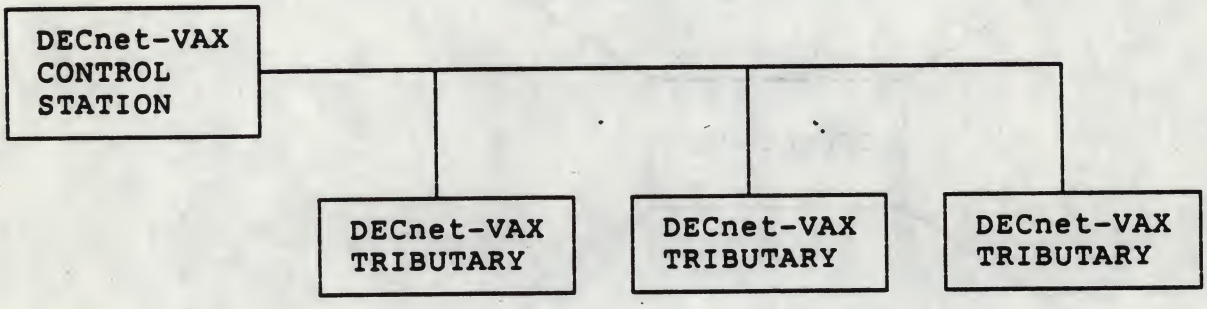


Multipoint

Multipoint connections allow one line to be shared by more than two nodes. Each node communicates over the line using a separate circuit. One line node, called the control station, controls access to the shared communications path; the other line nodes are called tributaries.

Because the line is shared by more than two nodes, an access control mechanism must determine which node can use the line at any given time. This is known as polling.

Figure 3-2: Multipoint Connection

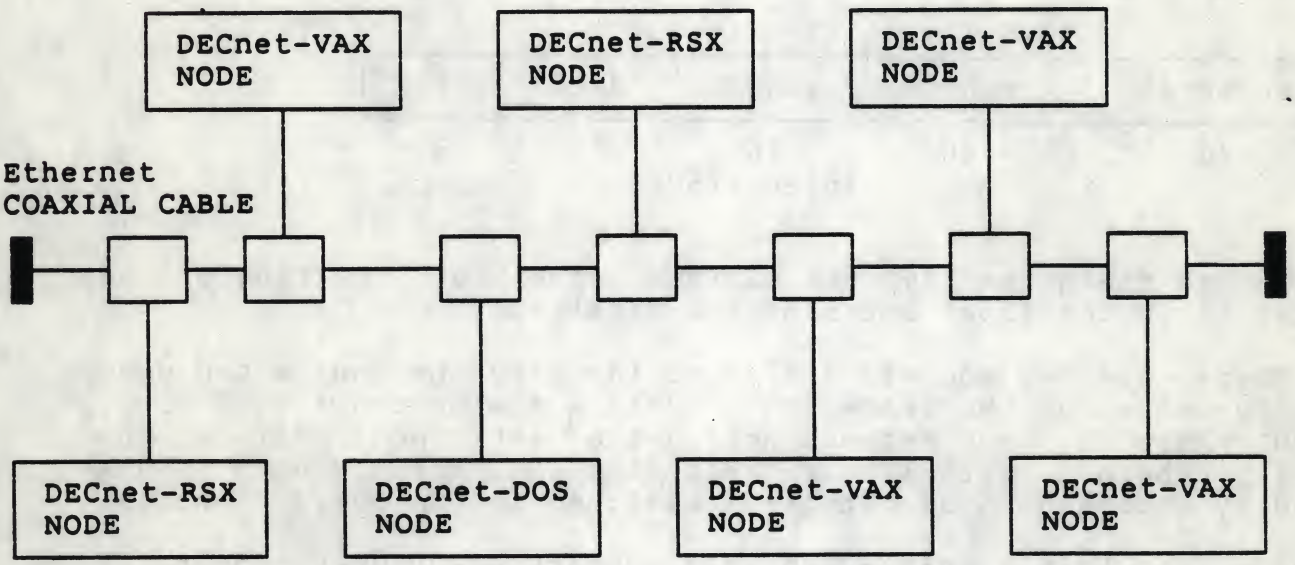


Ethernet

Operation

The Ethernet provides another means of connecting nodes to form a network. The Ethernet is a single communications bus that allows connection of up to 1,023 nodes. The bus structure of the Ethernet allows each node to have equal access to the communications channel. An access control mechanism is required to ensure that only one node transmits at a time. The technique used by the Ethernet is called Carrier Sense, Multiple Access with Collision Detect (CSMA/CD). In this method, each node can sense when the channel is being used. The node waits until the channel is free to transmit a message. Should nodes happen to transmit at the same time, each node will detect a message collision and wait a random time before retransmitting the same message.

Figure 3-3: Ethernet Configuration



Protocol

The Ethernet-module resides in the Datalink layer of DNA. The Ethernet Data Link layer provides a best effort delivery service. It does not provide an error-control facility to recover from transmission errors.

This layer has one type of message, the frame.
The Ethernet-frame format is:

destination	source	type	data	fcs
48	48	16	8n 46<=n<=1500	32

64 bit synchronisatie patroon

The new standard for Ethernet is IEEE 802.3. The format of the datapacket in IEEE 802.3 is slightly different from the Ethernet packet format. The IEEE 802.3 format is:

destination	source	length	data	fcs
48	48	16 46<=n<=1500	8n	32

meest optimaal

nn 2.8km max lengte

The length field can have the maximum value 1500. The type of the packet is in the first bytes of the datafield.

There are two address fields to identify the source and destination nodes of the frame. Both address fields contain 48 bits. In DNA Phase IV, each network node has a 16-bit node address. The 48-bit Ethernet data link address of a DNA Phase IV node is derived by attaching a 32-bit prefi assigned to DNA Phase IV nodes.

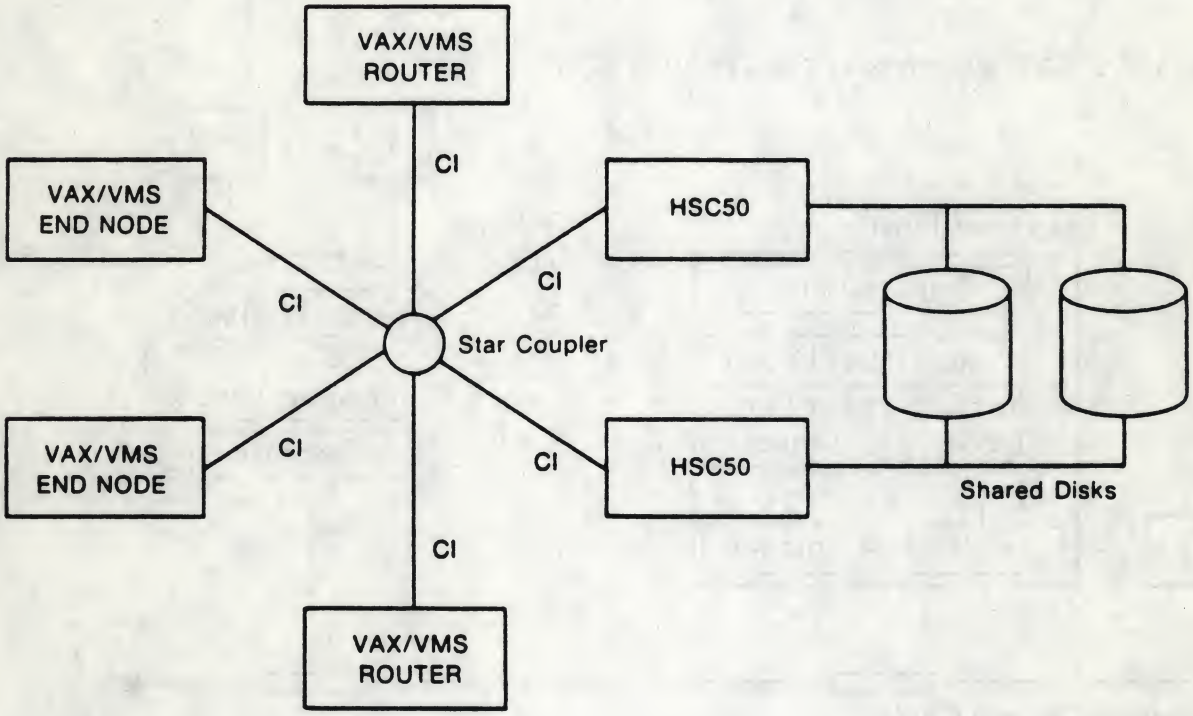
A node can have a unique Ethernet address 08-00-2b-03-45-26. If DECnet is running the node will use an other address, reflecting the DECnet node address. The format will be AA-00-04-00-XX-XX. The last four hexadecimal digits reflect the DECnet address. If the DECnet address is 30.132, the last four digits will be 84-78, so the address will be AA-00-04-00-84-78.

The type field is reserved for use by higher level protocols to identify the higher level protocol associated with the frame, permitting multiple higher-level protocols to coexist in the same Ethernet.

Computer Interconnect (CI)

The CI is a high-speed communications path that connects nodes in a VAXcluster configuration. VAXclusters consist of DECnet-VAX nodes and HSC50 or HSC70 disk controllers. Each individual node is connected by a CI cable to a Star coupler.

Figure 3-4: VAXcluster Configuration

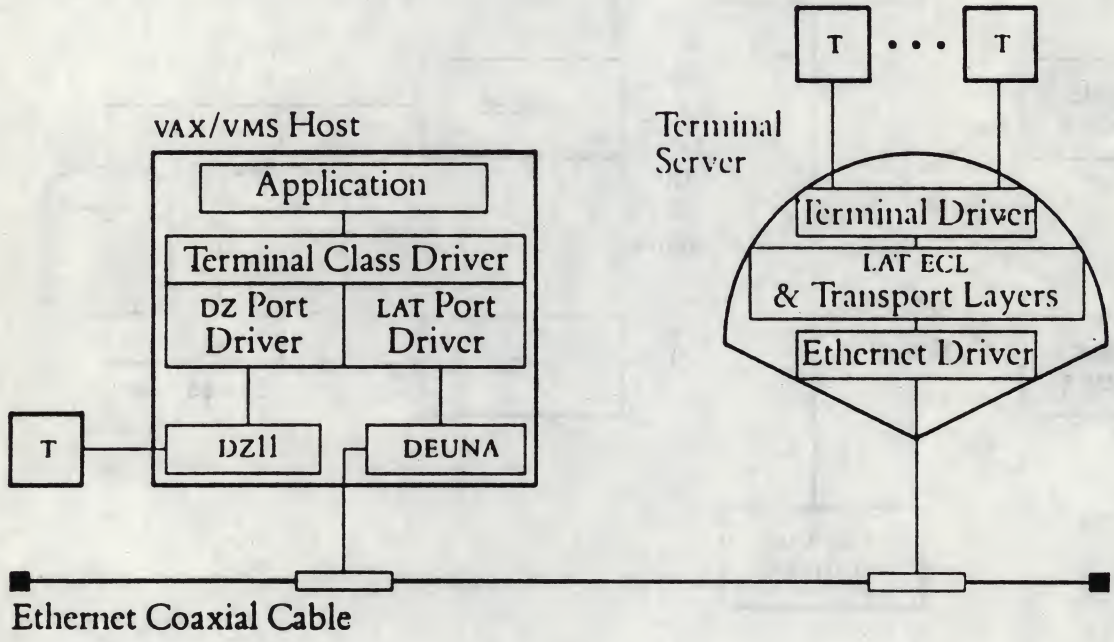


RELATIONSHIP ETHERNET - LAT

An Ethernet terminal server can be used to connect several users somewhere on the Ethernet to a connected host-node. In this situation the Local Area Transport protocol (LAT) is used.

LAT is an Ethernet-based virtual circuit protocol that does not use DECnet as a message transport facility. A terminal server on the Ethernet implements the LAT protocol to enable terminals that are connected to it to establish a logical link with any host node on the same Ethernet. The host node must implement the host side of the protocol.

Figure 3-5: LAT Protocol Implementation



Advantages of using terminal servers:

- **Less terminal processing on the nodes**
- **Increasing flexibility**
- **Higher availability of products by switching to another node**

The actions of the terminal server are transparent to users. You can use the terminal server in Local mode or in Service mode. In Local mode you interact with the terminal server, in service mode you interact with a computer on the network.

In Local mode you can use several commands. The most important are listed in table 3-1.

Table 3-1: Terminal Server Commands

Command	Description
HELP	displays information about the basic commands
HELP MORE	displays information about the commands to use the additional facilities
SHOW SERVICES	displays the services available to you
CONNECT XXX	initiates a session for the service XXX
[BREAK]	by pressing the break-key on the terminal you return to Local mode
SHOW SESSION	displays the sessions of your terminal
SHOW USERS	displays information about the connected terminals
RESUME x	reconnects you to your session nr x
DISCONNECT SESSION x	disconnects your session nr x
LOGOUT	disconnects all sessions and logs you out from the terminal server
SET FORWARD C	allows you to set up a special character to switch to the next session for your port

CONNECTING TO OTHER NETWORKS

Systems in a DECnet network can communicate with systems in non-DECnet networks. Specifically, DECnet nodes can exchange data and share resources with DECnet and non-DECnet nodes over an X.25 packet-switched data network (PSDN) and with systems in an IBM SNA network.

Connecting to an X.25 Network

A DECnet node can communicate over a PSDN with other DECnet and non-DECnet systems if it has been installed with Packetnet System Interface (PSI) software that enables the DECnet node to be connected logically to a DECnet Router/X.25 Gateway.

X.25 is a recommendation of the Comité Consultatif International Telephonique (CCITT), which defines a standard means for computers to interface with packet-switched data networks. A PSDN is a data communications service offered by common carriers, such as the Postal Telephone and Telegraph Authorities (PTT). Each PSDN consists of a number of geographically separated switching nodes that are connected by high-speed links. When an organization leases a circuit from the PTT or other carrier, the circuit physically connects a computer in the organization to one of the PSDN switching nodes, the data circuit terminating equipment (DCE). User Computers or terminals connected to DCEs are called data terminal equipment (DTE).

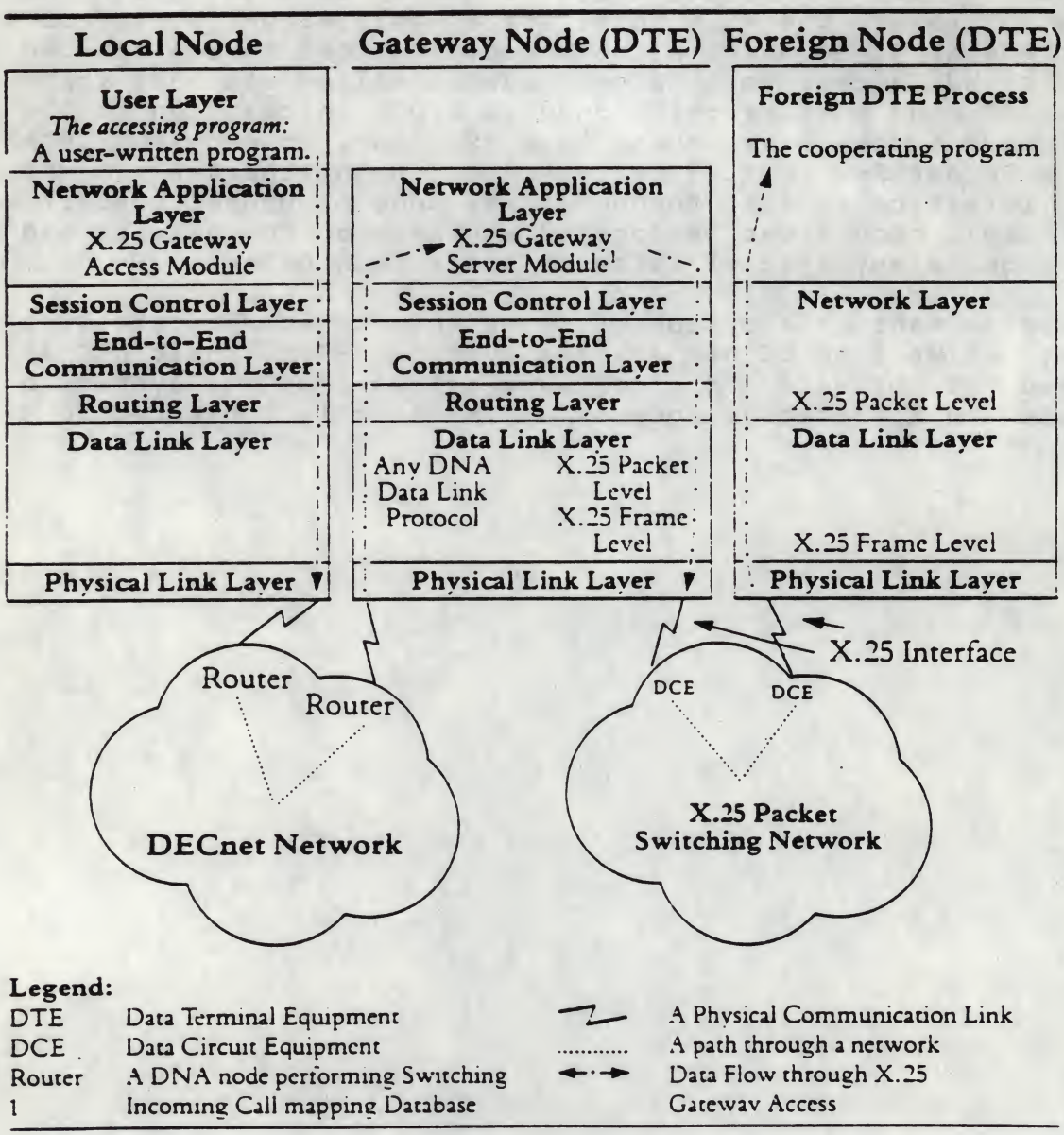
The way the network delivers the packets is completely transparent to the end user who has no influence over the path the packet takes through the network. The X.25 recommendation specifies the manner in which the data packets get from the user to the PSDN.

When a DECnet node wants to establish communication over a PSDN it has to run specific software and sometimes needs to be able to access dedicated hardware products.

If a DECnet node communicates with DECnet systems only, it may be able to use a DECnet facility called data link mapping (DLM). DLM allows two DECnet nodes to communicate as though the PSDN were not there. The X.25 interface in this situation is completely transparent to users at both nodes. DECnet node A can use DLM if it has access to a DECnet node installed with PSI software. The PSI node must be positioned as a DTE in relation to the PSDN. If node A does have access to a PSI node, it can send data through a DECnet Router/X.25 Gateway node on an Ethernet. To use the DLM interface (either through a PSI node or DECnet Router/-X.25 Gateway), node A can be located anywhere on the network and does not require any special software other than DECnet.

If users want the option of being able to communicate with non-DECnet as well as DECnet systems over the PSDN, their DECnet nodes need PSI software. This software allows a DECnet system to communicate in X.25 native mode if it is directly connected to a PSDN.

Figure 3-6: X.25 Gateway Access Operation



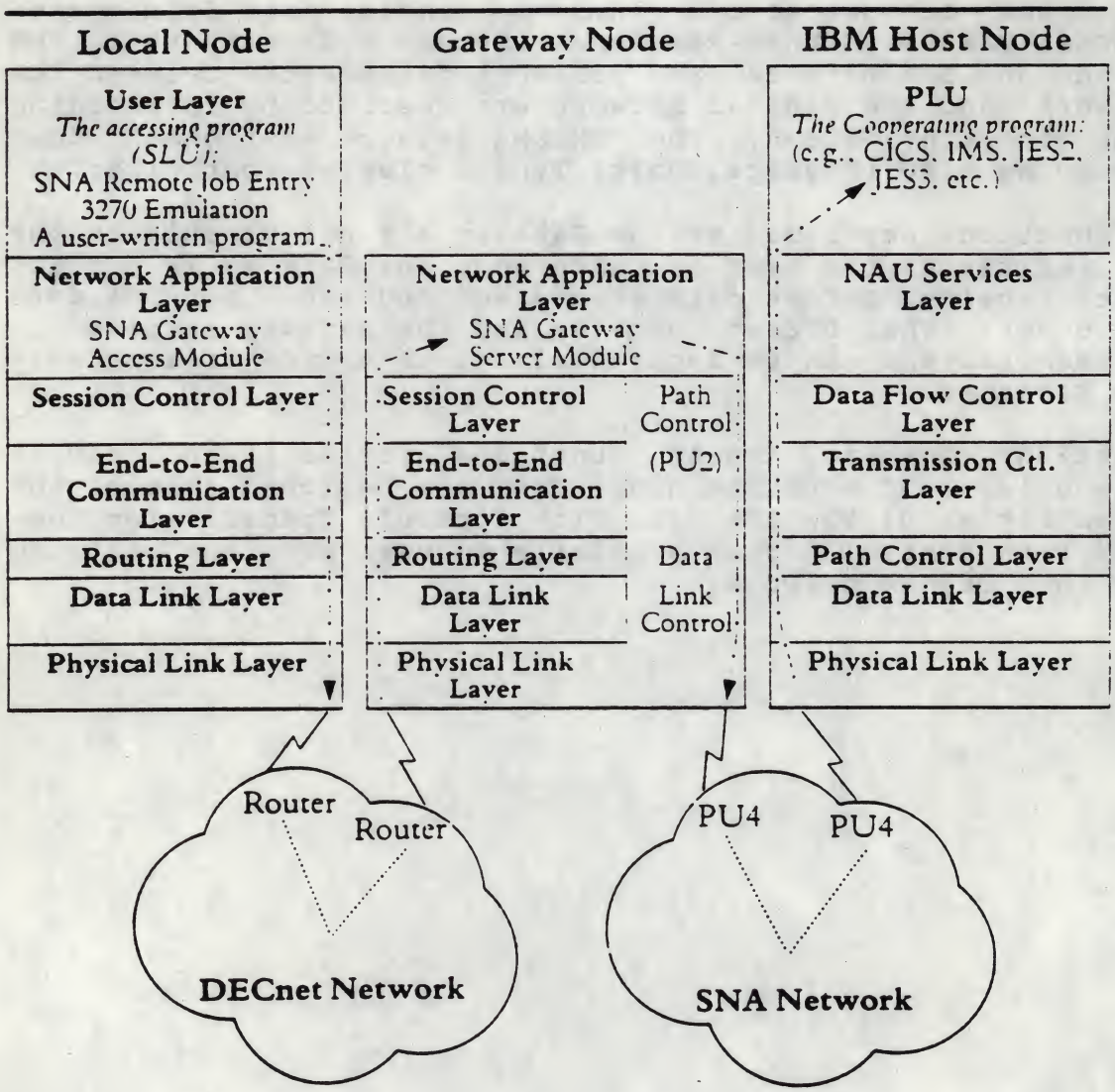
Connecting to an SNA-network

DECnet/SNA communications involve sending data from one or more DECnet-VAX or DECnet-RSX nodes through a Gateway to an IBM system in an SNA network. The protocol differences between the IBM network and the Digital network are resolved by interceding functions in the Gateway. The IBM SNA network environment sees the Gateway as a PU (Physical Unit) Type 2 cluster controller.

The functions performed at the Gateway are not visible to the end user. As far as the user is concerned, the Gateway is a black box that receives DECnet data at one end and sends out SNA data at the other. The DECnet node(s) and the Gateway involved in these communications can be located either in a wide area network or on an Ethernet.

DECnet/SNA Gateway Access functions reside in the Network Application layer of a DECnet node. They are designed to make the full capabilities of the SNA Data Flow Control, Transmission Control, and Path Control layers available to user programs residing anywhere in a DECnet network.

Figure 3-7: DECnet/SNA Gateway Access Operation



Legend:

SLU	Secondary Logical Unit	Router	A DNA node performing Switching
PLU	Primary Logical Unit	⟋⟍	A Physical Communication Link
PU2	Physical Unit Type 2	.....	A path through a network
PU4	Physical Unit Type 4	←·→	Data Flow through SNA Gateway Access

COMMUNICATIONS HARDWARE

To connect two computersystems you need some pieces of hardware.

There are a lot of interfaces you can use to make the physical connection to another system.

The most important interfaces you can use with DECnet-VAX are listed in table 3-2.

Table 3-2: Interfaces used for DECnet

Type	Description
DMC11	Synchronous interface, uses an old version of the DDCMP-protocol; replaced by the DMR11
DMR11	Synchronous interface; uses the DDCMP-protocol
DMP11	Synchronous interface; uses the DDCMP-protocol, can be used in multipoint and point-to-point mode
DMV11	Similar to the DMP11
DMF32	Interface with asynchronous DDCMP-lines and one synchronous DDCMP-line
CI	Computer interconnect interface
DEUNA	Ethernet-interface for Unibus systems
DELUA	New version of the DEUNA
DEQNA	Ethernet-interface for Q-bus systems
DELQA	New Version of the DEQNA
DESVA	Ethernet-interface for the VAX station
DECNA	Ethernet-interface for the PRO-350 and PR-380
DEBNA	Ethernet-interface for BI-bus systems

LINES AND CIRCUITS

In DECnet a LINE provides physical communications and is the lowest level communications path; it is part of the physical Link Layer.

Logical Link

A CIRCUIT is a high-level communications data path between nodes. A circuit operates over a physical line using one of the protocols DDCMP, CI, Ethernet and X.25.

For DDCMP, CI and Ethernet configurations each circuit is directly related to a corresponding line. For X.25 configurations the circuits and lines do not correspond directly. X.25 circuits are multiplexed to lines owned by the X.25 protocol handler module.

Line Identification

When the communication interface is used for point-to-point or multipoint connections the line identification takes one of the following formats:

dev-c

dev-c-u

where:

dev	is a communications interface device name
c	is a decimal number (0 or positive integer) designating the device's hardware controller
u	Is a decimal unit or line number (0 or positive integer) included if the device is a multiple unit line controller

Because the DMR11 is identical in operation to the DMC11, the lines have the same mnemonic for identification: DMC. For the same reason is the mnemonic DMP used for a DMV11 and a DMP11.

For the ethernet devices the line identification has the format:

dev-c

where dev signs to the controller used. We use the following mnemonics:

UNA-c	DEUNA or DELUA
QNA-c	DEQNA or DELQA
CNA-c	DECNA
SVA-c	DESVA
BNA-c	DEBNA

Circuit Identification

Each line has an associated circuit defined for software purposes. Point-to-point and Ethernet connections define one circuit per line; a multipoint connection defines as many circuits as there are connections.

The circuit identifications are:

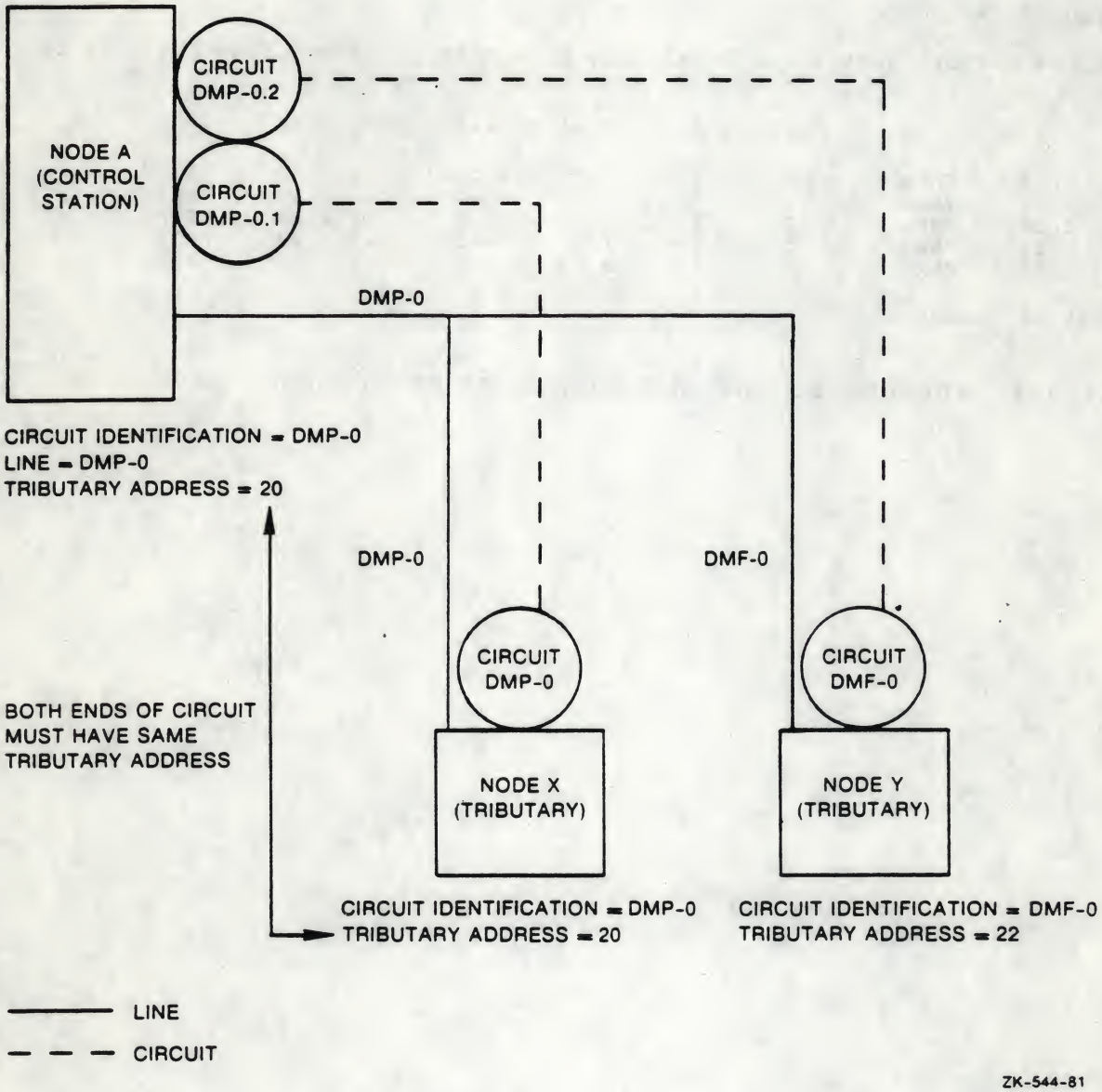
dev-c.x

dev-c-u.x

Where:

dev	is the circuit mnemonic
c	is a decimal number (0 or positive integer) designating the line
u	is a decimal line number (0 or positive integer) included if the device is a multiple unit line controller
x	is a decimal circuit number if the circuit is using a multipoint connection

Figure 3-8: Multipoint Circuits and Associated Lines



When we use the DCL-command SHOW NETWORK we see the circuits available from our node. See example 3-1.

```
$ SHOW NETWORK

VAX/VMS Network status for local node 2.13 VAXA on 12-AUG-1987 10:44:03.00

      Node          Links  Cost  Hops   Next Hop to Node

    2.13   VAXA       0     0     0     (Local)    -> 2.13    VAXA
    2.14   VAXB       0     3     1     UNA-0      -> 2.14    VAXB
    2.16   PDPA       0     5     1     DMC-0      -> 2.16    PDPA
    2.23   PROB       1     3     1     UNA-0      -> 2.23    PROB
    2.24   VAXC       0     8     2     UNA-0      -> 2.14    VAXB

Total of 7 nodes.
```

Example 3-1: The use of the DLC SHOW NETWORK command

Table 3-3: LINE and CIRCUIT States

Status	Line	Circuit
On	One of the circuits using this line must be on or on-starting	The circuit to the adjacent node is available
On-starting	Not applicable	The local node has been activated, but the remote node did not respond yet
Service	Used for testing purposes	Used for testing purposes
Off	None of the circuits using this line is in status on	The circuit has not been activated
Cleared	The protocol software has not been loaded	Not applicable
Synchronizing	Not applicable	The local node has been activated and is handling the start sequence with the remote system.

SUMMARY

To connect a system to a network, we need an interface and a protocol to control the datatransport.

Within DECnet we can choose the CI-protocol, the DDCMP-protocol, the Ethernet-protocol or the X.25-protocol.

To use these protocols we need special interfaces

The circuits in DECnet are displayed with a name derived from the name of the used interface.

The names of the circuits will be displayed when we use the DCL-command SHOW NETWORK.

USING THE NETWORK CONTROL PROGRAM

USING THE NETWORK CONTROL PROGRAM

INTRODUCTION

The Network Control Program is the utility program to use to configure, monitor and control the network. You can use NCP to display the reachable nodes, the available circuits and lines, the number of bytes sent or received. Using NCP in an effective way can give you a good insight in the network and its reliability.

OBJECTIVES

At the end of this module the student will be able to use NCP to display the characteristics of the nodes, lines and circuits.

RESOURCES

VAX/VMS Networking Manual
VAX/VMS NCP Reference manual

USING THE NETWORK CONTROL PROGRAM

To display information about the components that make up the network we use the Network CONTROL Program (NCP). This information can be derived from the volatile database.

The volatile database, maintained by NETACP, is a memory resident database containing current network configuration parameters.

The seven network components you can control using NCP are:

- Nodes
- Modules
- Circuits
- Lines
- Objects
- Logging
- Links

To activate NCP use the command:

```
$ RUN SYS$SYSTEM:NCP
NCP>
```

With HELP you get online information (see Appendix A)

To read the volatile database we use the NCP SHOW command. To change or set a parameter in the volatile database we use the NCP SET command.

To clear network parameters or reset them to the default values we use the NCP CLEAR command.

To use these NCP commands we sometimes need some privileges.

For the NCP SHOW command no special privilege is required; for the NCP SET and CLEAR commands the OPER privilege is needed.

To display the characteristics of the node components we use the command:

NCP SHOW <component> <parameter> <qualifier>

where:

<component>	EXECUTOR
	ACTIVE NODES
	KNOWN NODES
	ADJACENT NODES
	NODE XXX
	ACTIVE CIRCUITS
	KNOWN CIRCUITS
	ACTIVE LINES
	KNOWN LINES
	KNOWN LINKS
	LINE line-id
	CIRCUIT circuit-id
	LINK link-id
<parameter>	CHARACTERISTICS
	COUNTERS
	STATUS
	SUMMARY
<qualifier>	ADJACENT NODE node-id
	TO file-spec

With VMS V5 it is possible to use wildcards in NCP-commands.
For instance: SHOW NODE
or: SHOW circuit DMB_* CHARACTERISTICS.REM*

```
NCP> SHOW KNOWN LINES

Known Line Volatile Summary as of 12-AUG-1987 12:58:08

Line                  State

DMC-0                  on
DMF-0                  on
DMF-1                  on
UNA-0                  on


NCP> SHOW CIRCUIT DMC-0 CHARACTERISTICS

Circuit Volatile Characteristics as of 12-AUG-1987 11:43:22

Circuit = DMC-0

State                         = on
Substate                      = -starting
Service                       = enabled
Cost                          = 5
Hello timer                   = 15
Verification                  = disabled


NCP> SHOW ADJACENT NODES TO NODES.FILE

NCP> SHOW EXECUTOR SUMMARY

Node Volatile Summary as of 13-AUG-1987 11:57:51

Executor node = 2.13 (VAXA)

State                         = on
Identification                = DECnet-VAX V4.5,  VMS V4.5

NCP> SHOW ACTIVE NODES

Active Node Volatile Summary as of 13-AUG-1987 12:45:27

Executor node = 2.13 (VAXA)

State                         = on
Identification                = DECnet-VAX V4.5,  VMS V4.5


Node              State      Active  Delay   Circuit     Next node
Links

2.14   (VAXB)     reachable                  UNA-0         2.14 (VAXB)
2.16   (PDPA)     reachable                  UNA-0         2.16 (PDPA)
2.23   (PROB)     reachable                  UNA-0         2.23 (PROB)
2.24   (VAXC)     reachable                  UNA-0         2.14 (VAXB)
```

Example 4-1: Use of NCP

REMOTE USE OF NCP

You can use the NCP TELL-command to have the NCP-command executed at another node. The TELL-command sets the executor for only one command and must prefix the command for which it is intended.

```
NCP> TELL VAXA SHOW KNOWN NODES

NCP> TELL 2.45 SHOW EXECUTOR CHARACTERISTICS
```

Another way to have your NCP-commands executed at another node is setting the executor for your own process.

```
NCP> SET EXECUTOR NODE PDPA

NCP> SHOW KNOWN NODES

NCP> SHOW CIRCUIT UNA-0 CHARACTERISTICS

NCP> CLEAR EXECUTOR NODE
```

SUMMARY

The network Control Program is the utility program to use to configure, monitor and control the network. You can use NCP to display the reachable nodes, the available circuits and lines, the number of bytes send or received by using the NCP command NCP SHOW <component>.

ROUTING

INTRODUCTION

When a packet is sent from one system to another there has to be a path through the network. The systems the packet visits has to determine in which way this packet has to travel to its destination. This process is the routing process.

This module introduces the routing process, the operations performed and the terms used to describe the routing process.

OBJECTIVES

At the end of this module the student will be able to:

- Define and illustrate the terms associated with the DNA routing concept
- Describe the functional operations performed by the DNA's Routing Layer

RESOURCES

VAX/VMS Networking Manual
DECnet DIGITAL Network Architecture (Phase IV) General Description

ROUTING

DECnet implements a feature called adaptive routing which eliminates the need for every node to be physically connected to every other node in the network. Adaptive routing enables a message to get from the source node to the destination node even if it must pass through several intervening nodes.

Adaptive routing uses an algorithm that enables network nodes to send messages over the path that is the least costly in terms of system resources. In the event the chosen path fails, adaptive routing finds an alternative path to complete the communication.

DECNET NODE TYPES

DECnet supports two types of network nodes:

- Full function nodes or routing nodes
- End nodes or non-routing nodes

LEVEL I
LEVEL II

Full Function Nodes

Full function nodes (also called routing nodes) are nodes that can receive and forward messages addressed to other nodes in the network. This ability requires that the node possess and maintain a database listing of all the available paths to other network nodes. This database contains the cost of each alternative path to a destination node. It is updated by processing routing messages sent throughout the network when a node's routing information changes. The DECnet routing layer uses this information to perform the adaptive routing function.

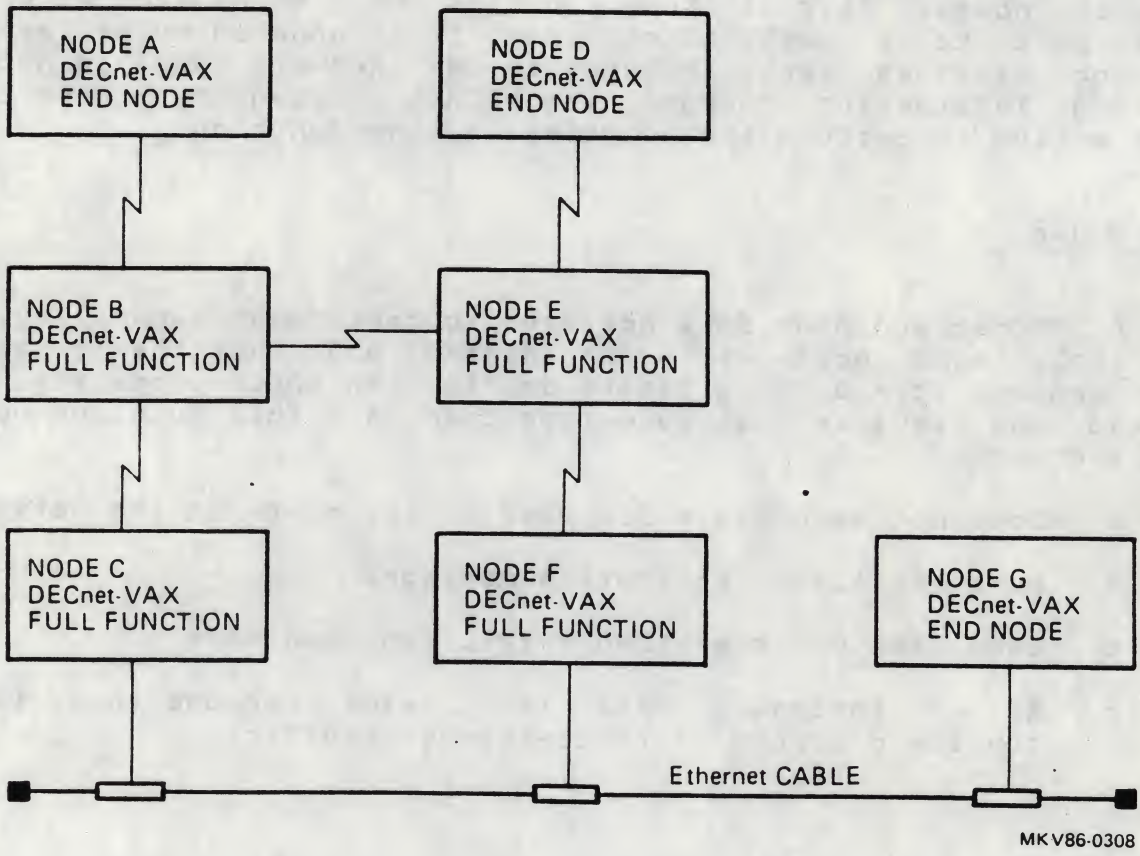
End Nodes

A DECnet end node does not have route-through capability. It can only send messages to one adjacent node that then forwards the message toward its ultimate destination. DECnet operation on an end node requires less resources than on a full function node. The end node:

- Does not maintain a database on the state of the network
- Does not listen to routing messages
- Uses less CPU time than a full function node
- Allows increased data link usage because there is no routing overhead or route-through traffic

Figure 5-1 shows an example of a network containing both full function nodes and end nodes.

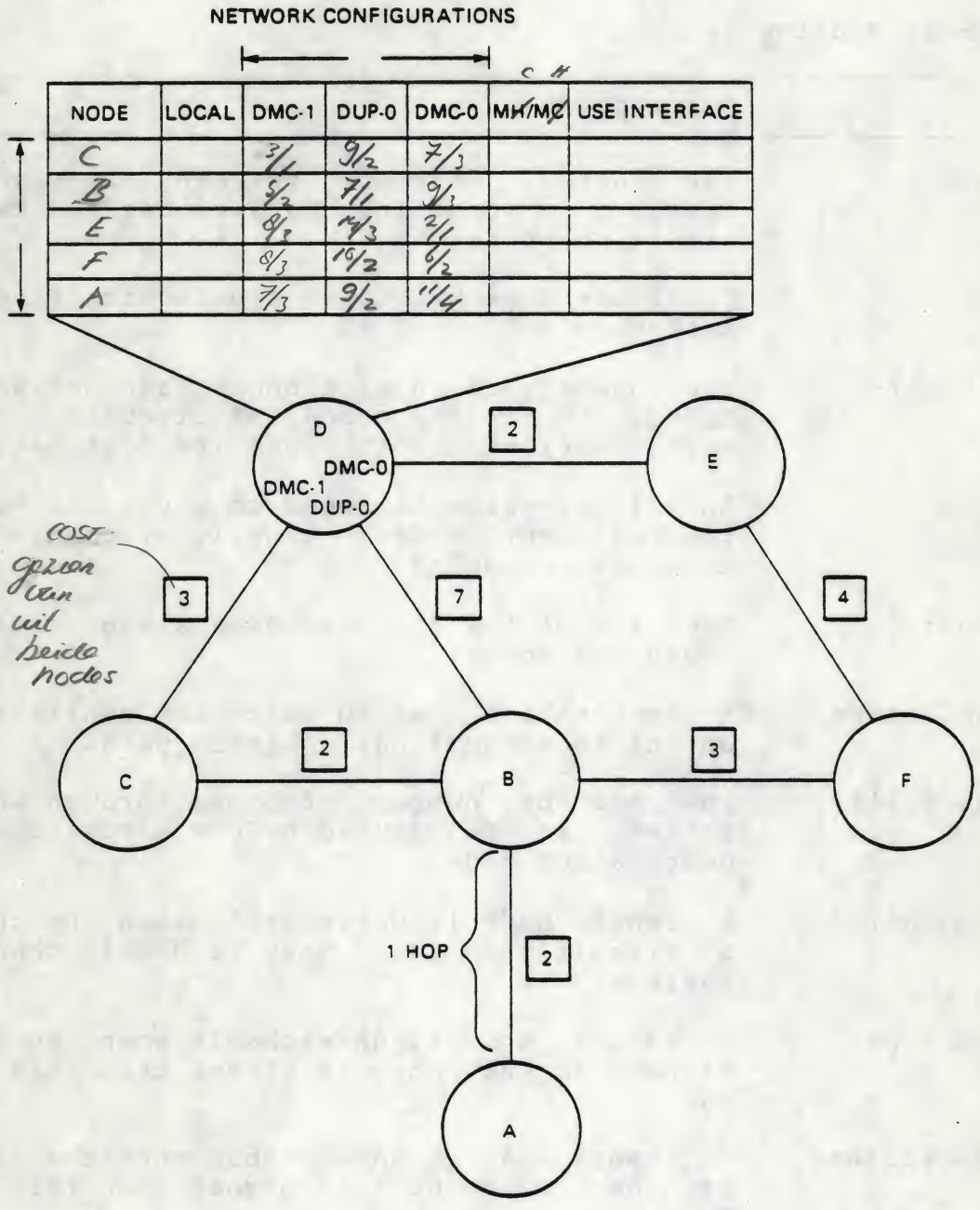
Figure 5-1: Sample DECnet Network



Using the network topology shown in figure 5-1 we can construct the routing databases for the different nodes.

Figure 5-2: Local Node Route Database Table

Route database built in a DECnet node during route initialization



ROUTING CONCEPTS AND TERMS

In table 5-1 the most important terms used with routing are listed.

Table 5-1: Routing Terms

Term	Description
Hop	The logical distance between two nodes is measured in hops. The distance between two adjacent nodes is one hop.
Path	The route a packet takes from source to destination.
Path length	The number of hops along a path between two nodes; it is the number of circuits a packet must travel across to reach its destination.
Cost	An integer value assigned to a circuit between two adjacent nodes, usually proportioned to transmission delay.
Path cost	The sum of the circuit costs along a path between two nodes.
Reachable node	A destination node to which the routing layer on the local node has a usable path.
Maximum visits	The maximum number of nodes through which a packet can be routed before arriving at the destination node.
Maximum cost	A remote node is unreachable when the cost of a circuit to that host is higher than this maximum.
Maximum hops	A remote node is unreachable when the number of hops to that host is higher than this maximum.
Maximum address	A remote node is unreachable when the address of the remote node is higher than this maximum.

Table 5-1: Routing Terms (continued)

Term	Description
Hello timer	Defines the frequency of router layer Hello messages
Listen timer	Controls the maximum amount of time allowed to elapse before the routing layer stops waiting for a message from the adjacent node on the circuit.
Routing timer	Defines the frequency of routing messages to all adjacent nodes.
Pipeline quota	The maximum number of messages or packets that a transmitting node can send without waiting for individual acknowledgments for each successive message.
Broadcast routing timer	Defines the frequency of routing updates on an Ethernet.
Max broadcast nonrouters	If the local node is a routing Ethernet node, this parameter defines the maximum number of endnodes for which the local routing node will store routing information. The endnodes are in the same area on the Ethernet.
Max broadcast routers	If the local node is a routing Ethernet node, this parameter defines the maximum number of routing nodes for which the local routing node will store routing information. The routing nodes are in the same area on the Ethernet
Designated router	A routing node on an Ethernet which provides a router service for the end nodes on the Ethernet.

Maximum path splits

Indicates the mamimum number of equal cost paths for a given destination node among which the packet load may be split.

Path split policy

Specifies the policy for equal cost load splitting of network traffic. If the policy has the value INTERIM, then all traffic will be split over all equal cost paths. If the policy has the value NORMAL, then all traffic will be split equally over all equal cost paths to a destination node.

The most part of these parameters is displayed by the NCP command NCP SHOW EXECUTOR CHARACTERISTICS (See example 5-1). The circuit-related parameters are displayed by the NCP command NCP SHOW CIRCUIT xxx-n CHARACTERISTICS; the line-related by the equivalent command for lines.

```
NCP> SHOW EXECUTOR CHARACTERISTICS

Node Volatile Characteristics as of  5-NOV-1988 11:50:13

Executor node = 2.14 (VAXB)

Identification            = DECnet-VAX V5.0,  VMS V5.0-2
Management version        = V4.0.0
Incoming timer            = 45
Outgoing timer            = 60
Incoming Proxy            = Enabled
Outgoing Proxy            = Enabled
NSP version               = V4.0.0
Maximum links             = 32
Delay factor              = 80
Delay weight              = 5
Inactivity timer          = 60
Retransmit factor         = 10
Routing version           = V2.0.0
Type                      = routing IV
Routing timer             = 600
Broadcast routing timer   = 180
Maximum address           = 1023
Maximum circuits          = 16
Maximum cost              = 1022
Maximum hops              = 30
Maximum visits            = 63
Maximum area              = 63
Max broadcast nonrouters  = 64
Max broadcast routers     = 32
Maximum path splits       = 1
Area maximum cost         = 1022
Area maximum hops         = 30
Maximum buffers           = 100
Buffer size               = 576
Nonprivileged user id     = DECNET
Default access            = incoming and outgoing
Pipeline quota            = 3000
Alias incoming            = Enabled
Alias maximum links       = 32
Alias node                = 2.24 (CLUSTER)
Path split policy         = Normal
```

Example 5-1: Using NCP to display the executor characteristics.

AREA ROUTERS

DECnet Phase III supports a maximum of 255 nodes. Phase IV DECnet nodes support new features as area routing, which permits the configuration of large networks and Ethernet support. Phase IV DECnet can support up to 63 areas with up to 1023 nodes in each area.

In DECnet Phase IV there are two kinds of routers:

- Level 1 routers
- Level 2 routers

Level 1 routers route traffic within a single area of the network. Their routing database contains routing information for only the nodes within their area.

Level 2 routers are known as area routers. They are used to routing messages between the nodes in one area and the nodes in other areas. They maintain two routing databases, one containing the nodes in their own area and the other containing information about all the other level 2 routers on the network.

SUMMARY

In a DECnet network a packet chooses a path from th source to the destination. The systems the packet visits has to decide which path it has to ho. These systems are the so-called Routers. These routers have a database with information on how to get to the different systems in the network in the most efficient way. A system forces a packet to take a certain circuit by looking at the cost of the circuit and to total the number of hops it takes to get to the destination.

The executor node can decide to forward the packet or not, by looking at certain parameters the network manager has set in his database. These parameters will be displayed by using the command NCP SHOW EXECUTOR CHARACTERISTICS.

USING SET HOST, PHONE AND MAIL

USING SET HOST, PHONE AND MAIL

INTRODUCTION

To use the resources on another system without disconnecting our terminal from our own system we can set up an interactive session on the remote system by using the SET HOST command. To communicate with other users on the systems in the network we use the MAIL and PHONE commands.

In this module the following commands are discussed:

- SET HOST
- MAIL
- PHONE

OBJECTIVES

At the end of this module the student will be able to use the SET HOST command and the MAIL and PHONE commands, and understand what is happening when he uses these commands.

RESOURCES

VAX/VMS Mail Utility Reference Manual
VAX/VMS Phone Utility Reference Manual
VAX/VMS Networking Manual

LOGICAL LINKS

A logical link connects an image or program running on the local node with an image running on a remote node. The logical link is a temporary data path that exists until one of the two images terminates the connection.

UTILITIES INTERFACES

The utilities available to the VAX/VMS user may also be used across the network. The user may access these utilities through standard DCL commands. When the DCL command includes a node name in the file specification, network access is initiated. When the DCL command initiates network access, a logical link is established.

The following list of DCL commands initiates utilities that allow the user to access remote network nodes:

- Remote command terminal
  - SET HOST
- Terminal-to-terminal communication
  - PHONE
- Network mail
  - MAIL
- Remote resource sharing (Module 8)
  - PRINT/REMOTE
  - BACKUP
  - SUBMIT/REMOTE
- File manipulation (Module 8)
  - APPEND
  - CLOSE
  - COPY
  - CREATE
  - DELETE
  - DIRECTORY
  - OPEN
  - READ
  - TYPE
  - WRITE
- Device handling (Module 8)
  - ASSIGN
  - DEASSIGN
  - DEFINE

REMOTE COMMAND TERMINAL

To Start a Remote Terminal Session

To start a remote terminal session, a user invokes the SET HOST command and specifies a remote node. After a connection has been made, the user can access the resources of the remote system.

- SET HOST <nodename>
- Log on to the remote system

You can use SET HOST more than one time.

Using SET HOST a logical link has been created between the local process and REMACP on the remote system. The protocol used is CTERM. Now the user has an RT_device on the remote node.

Using SET HOST to an RSX-system creates a logical link between the local process and RTH.

Terminating a Remote Session

To terminate a remote session on a VAX system, do one of the following:

- Log out
- Press CTRL/Y two or more times

```
$ SET HOST VAXA


                    WELCOME TO VAXA

                    PLEASE LOG IN


Username: STUDENT
Password: (not echoed)

$
```

Example 6-1: Using the SET HOST command

PHONE

You can use the PHONE-command for terminal-terminal communication. If the person you want to phone is not available on the system you can leave a message by using the MAIL command in PHONE.

Phone can also be used to display the users on a system somewhere on the net. For this purpose we use the DIR command in phone.

The qualifiers and commands we can use during the PHONE session are listed in table 6-1 and table 6-2.

```
                              VAX/VMS Phone Facility
%DIR VAXA
------------------------------------------------------------------------------
Press any key to cancel the directory listing and continue.

Process Name     User Name        Terminal         Phone Status

HACKER           JOHNSON          VTA24:           available
PETER            PETER            VTA218:          available
Marty James      JAMES            VTA290:          available

3 persons listed.
```

Example 6-2: Using the PHONE DIR command

Table 6-1: PHONE Qualifiers

Qualifier	Function
/scroll	When the viewport is full, the text is scrolled up one line and new text goes to the bottom line. Scroll is the default.
/noscroll	New text is wrapped and appears on the top line of the viewport. This will improve response on slow speed terminals.
/switchhook="C"	This is the character which must be typed before each phone command. The default is %.
/viewportsize=n	This is the maximum number of lines in the viewport, including heading line and bottom line of dashes. The valid range is from 3 to 10.

Table 6-2: PHONE Commands

Command	Meaning
ANSWER	Answers the phone when the user receives a call.
DIAL	Places a call to another user.
EXIT	Exits from PHONE.
FACSIMILE	Includes the contents of a file in the user's conversation.
HANGUP	Hangs up the user's own phone.
HELP	Displays information on how to use PHONE.
HOLD	Places other users on hold.
MAIL	Sends a short message to another user.
REJECT	Rejects a call from another user.
UNHOLD	Reverses the previous hold command.

MAIL

To send network mail to another person somewhere on the net you can use the MAIL facility. You can use the MAIL facility to send a message or a file to a single person, or a group of persons by using a distribution list.

Table 6-3: Summary of MAIL Commands

Command	Meaning
BACK	Backs up to the previous message.
COMPRESS	Creates a new MAIL.MAI file, without the deleted mails, and renames the old MAIL.MAI file to MAIL.OLD.
DELETE n-m	Deletes the messages.
DIRECTORY	Lists a summary of user messages.
EXIT	Exits form MAIL.
EXTRACT	Copies current message into a specified file.
FILE	Copies current (last-read) message into a specified folder.
FORWARD	Forwards current (last-read) message to user or users.
HELP	Displays information on how to use MAIL.
PRINT/QUE=XXX	Submits the current (last-read) message to queue=XXX
READ and <RETURN>	Displays next page, next message, or (READ only) specified message.
READ/EDIT n	Displays specified message, using the editor.

REPLY	Sends a reply to the sender of the current (last-read) message.
SEARCH string	Searches MAIL file for the first match of string.
SET FORWARD	The result of this command is, that mail will be sent to the specified destination.
SHOW ALL	Displays the characteristics of your mail session.

```
$ MAIL

You have 2 new messages.

MAIL> SEND
To:     VAXA::PETERS
Subj:   NEXT MEETING
Enter your message below. Press CTRL/Z when complete,
or CTRL/C to quit:
        Next fryday will be a meeting in room 4.23
        Time: 14:00 hour
        ^Z
MAIL>
MAIL> SEND/EDIT
To:     H711DOC
Subj:   Planning 24 OCT 1988
.....
.....

MAIL>DIR
                                                          NEWMAIL
    # From                  Date          Subject

    1 VAXA::PETERS          14-oct-1988   Plan new products
    2 VAXWRK::JOHNSON       18-oct-1988   Meeting next fryday

MAIL>READ 1
    #1          18-oct-1988 11:51:50.57                   NEWMAIL
From:   VAXA::PETERS
To:     WILLIAM
Subj:   Plan new products

Hallo Bill,

The products listed in Catalogue will be available at the end of
this week.

Bye, John.

MAIL> PRINT

MAIL> READ 2
        .........
        .........
MAIL> DELETE

MAIL> EXIT
Job MAIL (que SYS$PRINT, entry 243) started
```

Example 6-3: Sending and Reading your Mail

```
$ MAIL

You have 2 new messages.

MAIL>READ 1
    #1          18-oct-1988 11:51:50.57                    NEWMAIL
From:   VAXA::PETERS
To:     WILLIAMS
Subj:   Plan new products

Hallo Bill,

The products listed in Catalogue will be available at the end of
this week.

Bye, John.

MAIL> SET COPY_SELF REPLY,SEND

MAIL> REPLY/EDIT
To:     VAXA::PETERS  <-------- will be filled in by the system
Subj:   RE: Meeting next fryday
        ...........
        ...........

New mail on node VAXA from VAXA::WILLIAMS

MAIL> SEND/EDIT
To:     VAXA::PETERS
Subj:   COPY_SELF EXAMPLE
        ........
        ........

New mail on node VAXA from VAXA::WILLIAMS
```

Example 6-4: Answering a Mail

```
$ MAIL

You have 2 new messages.

MAIL> DIR/FOLDER

Listing of folders in DISK:[WILLIAMS]MAIL.MAI;1
     Press CTRL/C to cancel listing
MAIL                                    NEWMAIL

MAIL> READ 1
        .........
        .........

MAIL> MOVE PLANNING
Folder PLANNING does not exist.
Do you want to create it (Y/N, default is N)? Y

%MAIL-I-NEWFOLDER, folder PLANNING created

MAIL> DIR/FOLDER
Listing of folders in DISK:[WILLIAMS]MAIL.MAI;1
     Press CTRL/C to cancel listing
MAIL                                    NEWMAIL
PLANNING

MAIL> SELECT PLANNING

MAIL> DIR

                                                        PLANNING
    # From                  Date          Subject
    1 VAXA::PETERS          14-oct-1988   Plan new products
```

Example 6-5: Using Folders to Store Mails

Using a Distribution List

First you create a distribution list in the editor or by using the CREATE command. Use the file extension .DIS. Include for all the usernames the node if necessary. Using @filename will send the message to all the people in the list.

```
$ CREATE LIS.DIS
!Software writers
JOHNSON
VAXA::CHARLES
LAWRENCE
VAXB::ASHLEY
^Z
$ MAIL
MAIL> SEND/EDIT
TO:     @LIS
SUBJ:   Monthly Meeting
.........
.........
.........
.........
^Z
```

Example 6-6: Using a Distribution List

Message Files

A message file is stored as an ASCII text in the default logon directory. It contains all the short mailmessages. The default message file is MAIL.MAI.

```
From:    HDV                                23-JAN-87       16:23
To:      @GROUP
Subj:    Change of location

Next Tuesday's revieuw meeting has been moved from Conference Room
A to Conference Room B (on the second floor). See you there.

From:    MENDOZA                            24-FEB-87       23:20
To:      READER
Subj:    Payroll Program
There is a new version of the Payroll Program im my directory:
[MENDOZA]PAYROLL.FOR;12
```

Example 6-7: Example of a MAIL Message File

SUMMARY

With the SET HOST command we can setup a session on a remote system, as if the local terminal is directly connected to the remote system. In this situation all input from your terminal will be send by the local system to the remote system. Two processes will be active for servicing your actions.

The MAIL-command can be used to send information with other users on systems somewhere in the network they can read at a later moment.

The PHONE-command can be used to communicate with other persons on systems somewhere in the network interactively.

CREATING NETWORK PROCESSES

CREATING NETWORK PROCESSES

INTRODUCTION

To activate a program on another system you need a process.

In this module the different types of processes, and how to create them are discussed.

OBJECTIVES

At the end of this module the student will understand the way a network process is created.

RESOURCE

VAS/VMS Networking Manual

CREATING NETWORK PROCESSES

Communicating programs like any other image that runs on a VAX/VMS node, must run within the context of a process. The process context defines the environment in which images execute. The characteristics of the process allow the VMS software to control access and allocate resources for any operation attempted by an image running in the process. These characteristics include:

- Resource quotas
- Privileges
- Accounting data
- Scheduling priority
- Identification fields
  - USER NAME
  - User Identification Code (UIC)
  - Process Name
  - Process Identification Number (PID)

Process Types

When a process is created, it is assigned a process type. Process types include:

- Interactive
- Batch
- Network

In this module we shall describe the network process and how to create it.

Access Control Information

A network process must be created at the target node to provide an environment in which the target image can run. DECnet-VAX uses access control information (ACI) to determine which account will provide the characteristics for the network process.

Access control information, which consists of a user name and a password, identifies an account at the target node.

Once the user name and password are determined at the target node, the DECnet-VAX software either passes the request to an existing network process associated with the proper account, or creates a new network process using the characteristics from the designated account.

The information included in the communication request is called outbound access control.

Default DECnet Accounts

DECnet-VAX provides the network system manager with the option of establishing default DECnet accounts that are used to allow communication with a network object without providing explicit access control information. Default DECnet accounts, like any user account, have entries in the SYSUAF.DAT configuration file. A default DECnet account may be either privileged or nonprivileged. The privileged account must be used when the requested target object requires privileges to perform its functions. When the object does not require privileges, the network process is associated with the nonprivileged account.

OUTBOUND ACCESS CONTROL INFORMATION

Three types of outbound access control information can be sent from the source node:

- Null access
- Explicit access
- No access

Null Access

Null access causes an ACI string of null characters to be sent to the target node. If a default DECnet account exists on the target node, the user can use null access to force a connection to this account.

```
$ DIR SNEEZY"":::DISK1:[PUBLIC]
```

The null string is sent to the remote node along with the process identification number (PID) of the source process. DECnet-VAX in the target node associates the nonprivileged default DECNET account with the incoming connection request. A network process is created with the privileges, priorities, and resource availability of this account ([DECNET]).

Explicit Access Control

User-supplied explicit ACI is implemented by including the target account's user name and password in quotation marks following the node specification in a command or a program statement. For example:

```
$ DIR SNEEZY"EARTH MARS":::DISK1:[EARTH]
```

When DECnet-VAX recognizes explicit access control, it will send the provided user name and password to the target node. The communication request also includes the PID of the source process to identify the process requesting the logical link.

No Access

The third type of outbound access control information is no access. When neither null access nor explicit ACI is used, DECnet-VAX attempts to use proxy access.

Proxy access is a method of establishing communication without sending access control information with the communication request.

When DECnet-VAX at the source node determines that proxy access is enabled, it will send no access control information and will send the user name of the requesting process instead of the PID.

When the user does not use null access or supply a user name and password, and the user does not have a proxy account, DECnet-VAX attempts to generate default explicit access control information. For example:

```
$ DIR SNEEZY::[PUBLIC]
```

DECnet-VAX attempts to identify a user name and password for a default DECnet account at the target node. The user name and password must be listed in an entry for the target node in the node database at the source node. The default account can be privileged or nonprivileged.

ESTABLISHING A LOGICAL LINK

DECnet-VAX allows two images, each running in a different process, to communicate with each other. A program that can communicate over the network is called an OBJECT.

For this communication to take place, the images must establish a connection to allow control and data messages to be exchanged. This connection is called a logical link.

Figure 7-1: The Logical Link

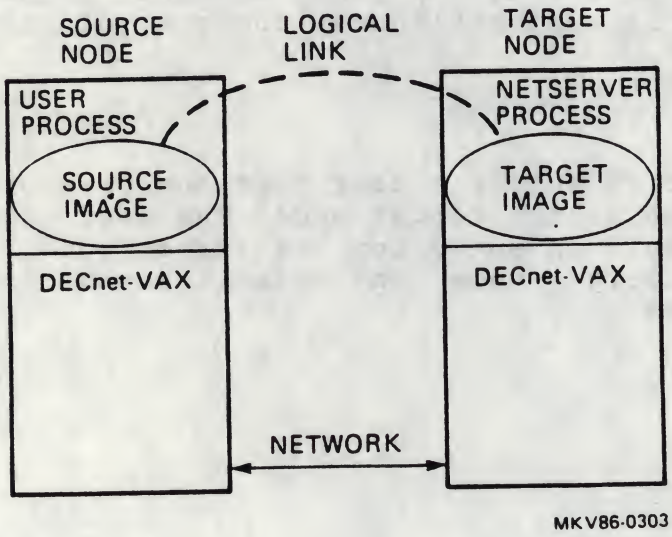


Figure 7-2: Establishing the Logical Link (Outbound)

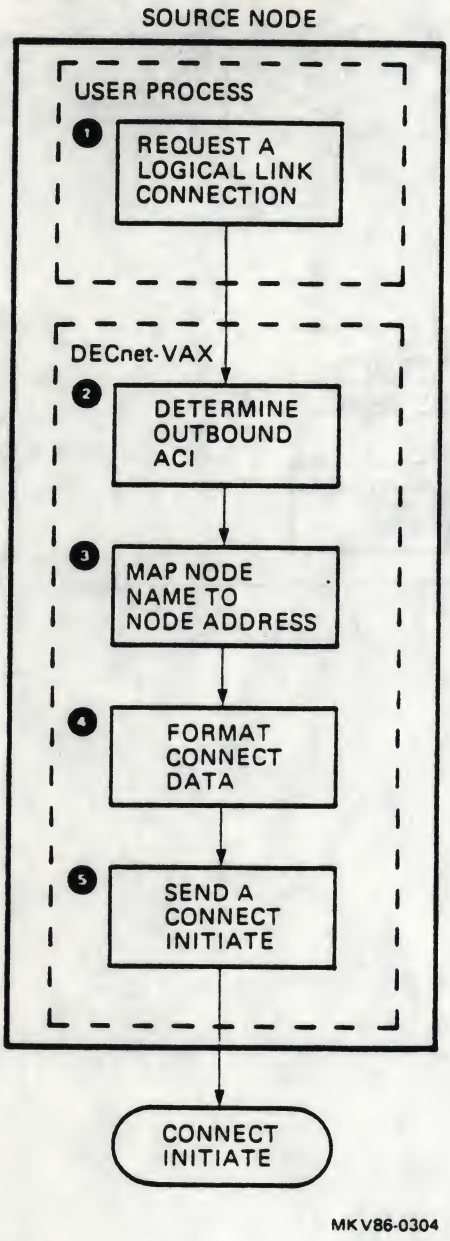
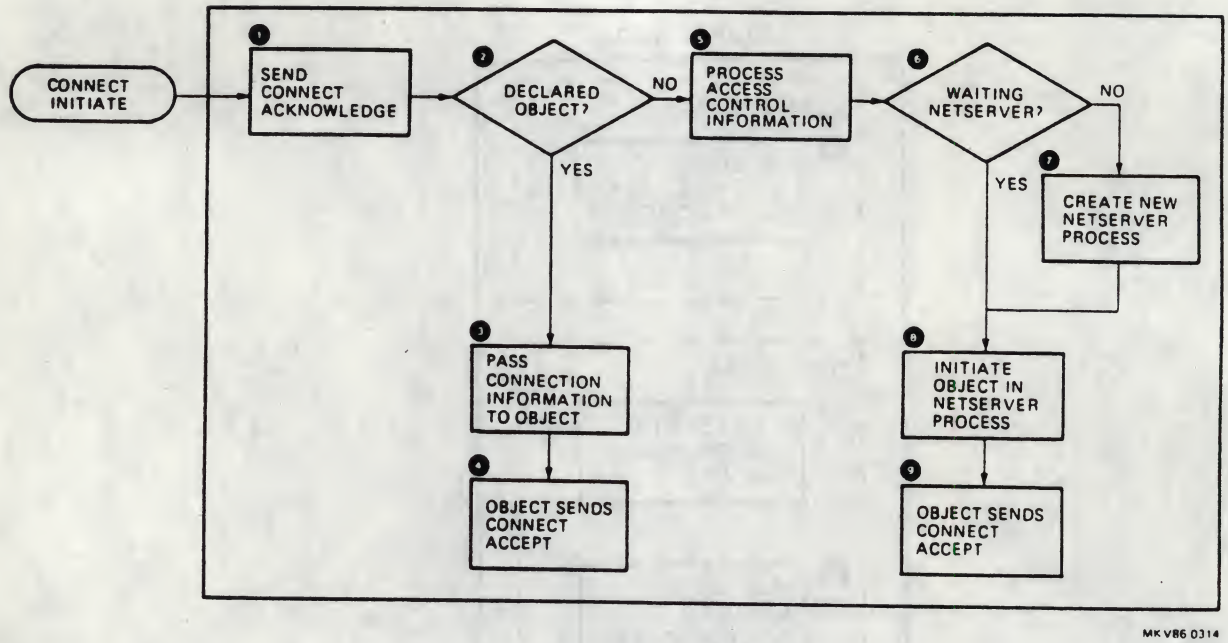


Figure 7-3: Establishing the Logical Link (Inbound)



If there is no network process created to accept the logical link request, DECnet creates a new netserver process to handle the connection. The user name and password obtained by processing the inbound access control information identify the account to by used to determine the attributes of the new network process. The following steps describe the netserver process creation.

- DECnet issues a $CREPRC system service call, which passes control to LOGINOUT.EXE

- LOGINOUT performs the following functions:

  - opens SYSUAF.DAT
  - creates a process
  - creates a logfile
  - passes control to DCL

- DCL performs the following functions:

  - executes system login command file
  - executes user's login command file
  - passes control to NETSERVER.COM

- NETSERVER.COM starts NETSERVER.EXE

After the logical link is disconnected, the NETWORK process in which it was running becomes idle. The netserver process remains in this wait state for a period of time defined by a time-out parameter called NETSERVER$TIMEOUT; after that period it is deleted.

SUMMARY

A NETWORK process will be created, when we activate a program on another system in the network. to create this NETWORK-process the system uses the user-specified login-information, a username and a password.

The remote system uses the username to create a NETWORK process with the characteristics specified in the SYSUAF.DAT file.

When the user does not specify login information, the NETWORK process will be created on account of the DECNET-account (if there is any).

After a specified period of inactivity the NETWORK-process will die.

REMOTE FILE ACCESS

REMOTE FILE ACCESS

INTRODUCTION

In this module will be discussed how to use the different DCL commands to access files on remote systems. This includes how to print a file on a remote system, how to execute a file on a remote system, and how to open and close a file on a remote system.

OBJECTIVES

At the end of this module the student will be able to use the different DCL commands to access resources and files on remote systems.

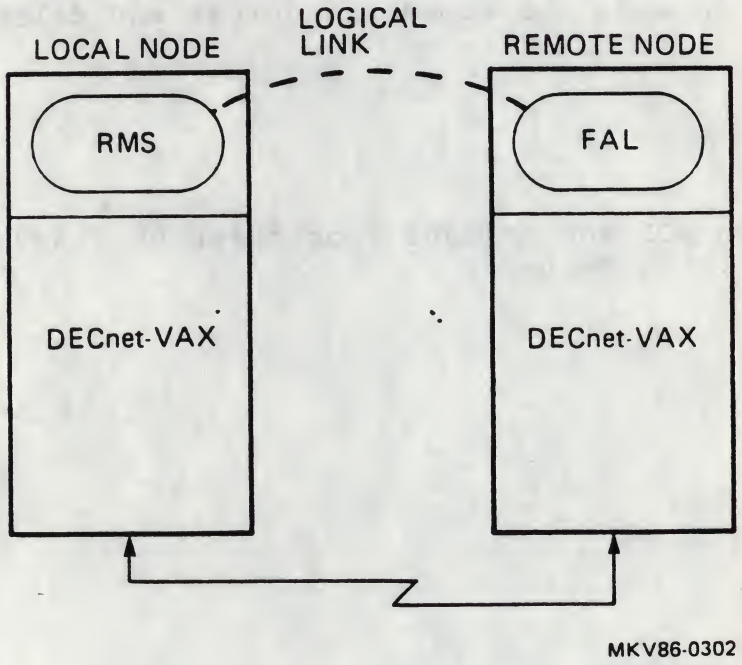
RESOURCES

Guide to Using DCL and Command Procedures on VAX/VMS
VAX/VMS Networking Manual

THE COMMUNICATING PROGRAMS

Remote file access is an example of two programs in different nodes communicating with each other. In Figure 8-1, the programs are RMS at the source node and the File Access Listener (FAL) at the target node.

Figure 8-1: Remote File Access Communicating Programs



RMS is the file and record subsystem of the VAX/VMS operating system. It provides the user with access to the file system on the local node as well as access to file systems at remote nodes.

FAL is a DIGITAL-supplied network object that provides authorized access to the file system of a DECnet node on behalf of processes executing on any node in the network. FAL resides in the Application layer of the DECnet architecture. The FAL object is always defined as object number 17 in the object database of a DECnet node.

THE NODE SPECIFICATION

Including a node specification in a command or program statement causes RMS to initiate remote file access through the DECnet-VAX software.

```
$ TYPE VAXA::DISK2:[ROMAN]TEST.FIL
```

The node specification may optionally include explicit access control information used at the remote node to determine the account used to create the netserver process in which the FAL object is run.

```
$ TYPE VAXA"ROMAN ROBOT"::DISK2:[ROMAN]TEST.FIL
```

This command causes the netserver process created at node VAXA to possess the attributes of the user account ROMAN.

VAX/VMS allows you to assign a logical name to represent the node specification in a DCL command or program statement. Using logical node names allows you to use a network node specification in a DCL command procedure or program statement without entering the specific node name and access control information.

An advantage of using a logical node name is security. You can access a specific account on a remote node without specifically stating the user name and password of that account in your program.

The following examples show the use of logical names to represent the node specification portion of file specifications.

```
$ DEFINE SWAN "SWANZY""ROMAN ROBOT"":"
```

```
$ TYPE SWAN::DISK2:[ROMAN]TEST.FIL
```

Logical node names can also be used in program statements as in the following FORTRAN OPEN statement.

```
$ DEFINE REMOTE_NODE "SWANZY""ROMAN ROBOT"":"
```

```
$ RUN SOURCEPROG
```

Where SOURCEPROG.FOR contains the statement:

```
OPEN (UNIT=1, NAME='REMOTE_NODE::TEST.FIL',....)
```

REMOTE FILE ACCESS USING DCL-COMMANDS

We can access files by using normal DCL-commands.

APPEND

```
$ APPEND VAXA"USER PASSWORD"::FILE1.TMP
_To: VAXB"USER2 PASSW"::FILE2.TMP
```

COPY

```
$ COPY FILE1.BASIC
To: PDPA"RSXUSER PASSWORD"::FILE.XYZ
```

DELETE

```
$ DELETE VAXA::*.TMP.*/LOG
```

TYPE

```
$ TYPE PDPA"USER PASSWORD"::NEXT.FILE
```

CREATE

```
$ CREATE VAX::FILE.NEXT
LINE1
LINE2
LINE3
^Z
```

DIRECTORY

```
$ DIRECTORY VAX::

Directory VAX::USE_DISK:[DECNET]

NETSERVER.LOG;2027    NETSERVER.LOG;2026    TEST.TST;1
TTJ1.DAT;1            TTK4.LIS;4            TTK4.LIS;3

Total of 6 files.
```

THE NETSERVER.LOG FILE

The logfile created during creation of the network process is NETSERVER.LOG.
This file contains a copy of the login command files executed during process creation, a copy of the NETSERVER.COM file, and information about each connection request received by the net-server process. The command sequence shown in example 8-1 leaves us the logfile shown in example 8-2.

```
$ DEFINE REVAXA "VAXA""USER PASSWORD"""::"
$
$ DIR REVAXA:*.TXT

Directory VAXA"USER password"::USE_DISK:[USER]

TEXT.TXT;1            EXPL.TXT;1

Total of 2 files.
$
$ TYP REVAXA:TEXT.TMP
This is the file text.tmp
mumble mumble mumble
^Z
$
```

Example 8-1: Using DCL Commands for remote file access

```
$       mode    =       F$VERIFY(0)
$
$loginfile:
$
$       mode    =       F$VERIFY(mode)
$       if      f$search("LOGIN.COM") .nes. ""  then @login
$! LOGIN.COM
$!
$! My user-LOGIN.COM
$!
$       IF ( F$MODE() .EQS. "NETWORK" ) THEN GOSUB NET
$ NET:
$       SHOW PROCESS

14-AUG-1987 11:03:09.20                         User: USER
Pid: 20400B5E   Proc. name: FAL_3097            UIC: [LOCAL,USER]
Priority:   4   Default file spec: USE_DISK:[USER]
$       RETURN
$       IF ( F$MODE() .EQS. "BATCH" ) THEN GOSUB BATCH
$       IF ( F$MODE() .EQS. "INTERACTIVE" ) THEN GOSUB INTER
$       EXIT
$ IF "" .NES. "" THEN NETSERVER$COMMAND
$ IF ""  .EQS. "" THEN NETSERVER$VERIFY = 0
$ V = F$VERIFY(NETSERVER$VERIFY)

        ------------------------------------------------------

        Connect request received at 14-AUG-1987 11:03:12.05
            from remote process VAXB::"0=JOHNSON     "
            for object "SYS$COMMON:[SYSEXE]FAL.EXE"

        ------------------------------------------------------


        ------------------------------------------------------

        Connect request received at 14-AUG-1987 11:03:18.40
            from remote process VAXB::"0=JOHNSON     "
            for object "SYS$COMMON:[SYSEXE]FAL.EXE"

        ------------------------------------------------------

  USER    job terminated at 14-AUG-1987 11:08:20.17
  Accounting information:
  Buffered I/O count:          235      Peak working set size:   298
  Direct I/O count:            132      Peak page file size:    1265
  Page faults:                3028      Mounted volumes:           0
  Charged CPU time:     0 00:00:08.94   Elapsed time:     0 00:06:46.05
```

Example 8-2: NETSERVER.LOG file

REMOTE COMMAND PROCEDURE EXECUTION

To submit a command procedure on a remote node the specified procedure has to reside at one of the disks on the remote node.

Not all of the SUBMIT qualifiers are compatible with /REMOTE. Only the following qualifiers may be specified with /REMOTE: /BACKUP, /BY_OWNER, /CONFIRM, /CREATED, /EXCLUDE, /EXPIRED, /MODIFIED, and /SINCE.

When the command procedure executes, all output is written to a log file on the node and in the account where the file resides. By default the log file specification consists of the file name of the command procedure and the file type LOG; the file will be deleted after printing.

```
$ !
$ ! DCLTEST.COM - a command procedure used to demonstrate
$ ! DECnet remote command procedure execution
$ !
$ START:
$       SHOW PROCESS
$ END:
$ !
$       EXIT
```

Example 8-3: Command procedure DCLTEST.COM

```
$       mode    =       F$VERIFY(0)
$
$loginfile:
$
$       mode    =       F$VERIFY(mode)
$       if      f$search("LOGIN.COM") .nes. ""  then @login
$! LOGIN.COM
$!
$! This is my LOGIN.COM
$!
$       IF ( F$MODE() .EQS. "NETWORK" ) THEN GOSUB NET
$       IF ( F$MODE() .EQS. "BATCH" ) THEN GOSUB BATCH
$ BATCH:
$       RETURN
$       IF ( F$MODE() .EQS. "INTERACTIVE" ) THEN GOSUB INTER
$       EXIT
$ !
$ ! DCLTEST.COM - a command procedure used to demonstrate
$ ! DECnet remote command procedure execution
$ !
$ START:
$       SHOW PROCESS
13-AUG-1987 16:29:22.81                         User: USER
Pid: 20400AB9   Proc. name: BATCH_532           UIC: [LOCAL,USER]
Priority:   1   Default file spec: USE_DISK:[USER]
$ !
$       EXIT
  USER     job terminated at 13-AUG-1987 16:29:23.59
  Accounting information:
  Buffered I/O count:              38      Peak working set size:    337
  Direct I/O count:                39      Peak page file size:     1215
  Page faults:                    536      Mounted volumes:            0
  Charged CPU time:     0 00:00:02.27      Elapsed time:     0 00:00:27.49
```

Example 8-4: Logfile of the Batch command procedure DCLTEST.COM

REMOTE PRINTING

When printing a file on a printer on system AAA the file must be located on one of the disks of system AAA.
You always have to specify the qualifier /REMOTE. If this qualifier is specified, the file is queued for printing in the default queue of the node on which the file exists.

Not all PRINT qualifiers are compatible with /REMOTE. Only the following qualifiers may be specified with /REMOTE: /BACKUP, /BEFORE, /BY_OWNER, /CONFIRM, /CREATED, /EXCLUDE, /EXPIRED, /MODIFIED, and /SINCE.

```
$ PRINT/REMOTE VAXA::FILE3.TXT
```

If the file is not yet on the remote system you have to copy the file first, before printing it.

```
$ COPY FILE3.TXT VAXA::FILEREMOTE.TXT
$ PRINT/REMOTE VAXA::FILEREMOTE.TXT
```

ACCESSING REMOTE FILES USING DCL COMMAND PROCEDURES

DCL command procedures executing at the local node can contain DCL commands that access files residing on remote nodes. Like all commands that request remote file access, these commands must contain a node specification identifying the node where the file resides.

```
$ ! REMSEQFILE.COM  -   Command procedure to read all the record
$ !                     of a sequential file on a remote node and
$ !                     type them to the terminal on the local node
$ !
$ !
$       OPEN/READ REMOTE NOOT"USER PASSWORD"::TEST.FIL
$ !
$ READ_LINE:
$       READ/END_OF_FILE=EOF    REMOTE  LINE
$ !
$       WRITE SYS$OUTPUT LINE
$ !
$       GOTO READ_LINE
$ !
$ ! End of file routine
$ !
$ EOF:
$       CLOSE REMOTE
$       EXIT
```

Example 8-5: DCL Command procedure REMSEQFIL.COM

SUMMARY

The DCL-commands for file handling are also available for handling files at remote nodes. The file specification has to specify the nodename and a username/password to create a NETWORK process to handle the file at the remote system.

The file EXAMPLE.DAT on remote node VAXA in the directory [STUDENT] can be accessed by the filespecification VAXA"STUDENT PASSWORD"::[STUDENT]EXAMPLE.DAT;1

All activities at the remote node on account of the created NETWORK-process are logged in the file NETSERVER.LOG in the default directory of the specified account.

To use the PRINT and SUBMIT-command at a remote node you have to specify the REMOTE qualifier.

OBJECTS

OBJECTS

INTRODUCTION

In this module is discussed what kind of programs are used to communicate over the network. These programs are called OBJECTS.

OBJECTIVES

At the end of this module the student will be able to write a Zero-Object, a DCL-command procedure, and understand how this object can cooperate with a procedure on the local system.

RESOURCES

Guide to Using DCL and Command Procedures on VAX/VMS
VAX/VMS Networking Manual

OBJECTS

A program that can communicate over the network is called an object. Objects can be either user-written or supplied by DIGITAL. They are identified by object type (number) and name. Before an object can receive any communication requests, it must identify itself to the DECnet software on its node. The object parameters are included in an object database that is accessed by DECnet-VAX whenever it receives a communication request. The parameters that can apply to an object include:

- An object number
- The file to be executed when the object is requested
- Privileges required to connect to the object
- PROXY enabled or disabled
- Optional default access control for inbound connects

PROXY and access control information are discussed in module 7 of this course.

A source program uses an object type to identify the target object with which it wishes to communicate.

Objects are divided into two general types:

- Zero (0) Objects
- Nonzero Objects

User-written objects are most of the time not entered in the object database. They are objects in the category of the zero-objects. They are called by using the TASK-object, which is defined in the object database as a zero-object.

Nonzero objects are known objects that are identified in the object database by their object type number. Object type numbers for nonzero objects range from 1 to 255.

DIGITAL reserves the range of nonzero object numbers between 1 and 127 for DECnet system objects. The remainder of the nonzero object types (128-255) are available to identify user written programs.

Table 9-1: DIGITAL Supplied Object Types

Object Number	Object name
0	TASK
17	FAL - File access listener
18	HLD - Host loader
19	NML - Network management listener
23	REMACP
25	MIRROR - Loopback mirror
26	EVL - Event logger
27	MAIL
29	PHONE
42	CTERM
63	DTR - DECnet test receiver

EXAMPLE OF A ZERO OBJECT

A zero object can be a user written program or command procedure that uses the network. When we want two programs to communicate across the network a logical link must be created.

When a source program initiates a logical link request, it identifies the remote node and the remote task with which to communicate. The remote node is specified in the node specification string. Since the remote task is an object type 0, the object name must be supplied in the task specification string.

The task specification string is a quoted string that identifies the remote task to which you attempt a logical link connection. It must be included in the program statement used to initiate the logical link.

"TASK=taskname"

"0=taskname"

Note that 0 and TASK are equivalent. The task name can be from 1 to 16 characters and represents the filename of a command procedure to be executed in the netserver process at the remote node when the logical link is initiated.

```
$ !
$ ! SOURCE.COM
$ !
$       INQUIRE STRING "Enter string: "
$ !
$       OPEN/WRITE/READ LINK VAX"USER PASSWORD"::"TASK=TGT"
$ !
$       WRITE LINK STRING
$ !
$       READ LINK COUNT
$ !
$       WRITE SYS$OUTPUT COUNT
$ !
$       CLOSE LINK
$ !
$       EXIT
```

Example 9-1: DCL Source Program SOURCE.COM

```
$ !
$ ! TGT.COM
$ !
$       OPEN/WRITE/READ LINK SYS$NET
$ !
$       READ LINK STRING
$ !
$       COUNT = F$LENGTH(STRING)
$ !
$       WRITE LINK COUNT
$ !
$       CLOSE LINK
$ !
$       EXIT
```

Example 9-2: DCL Target Program TGT.COM

SUMMARY

A program used in a communication over the network is called an OBJECT. A user written.program, used to communicate with a program at another system can have a so called objectnumber, 128-255.

A network program supplied by DEC has an objectnumber 0-127.

To start an object at a remote node you have to specify the object name or number in the task specification string. For instance, to activate the command procedure EXAMPLE.COM at the node VAXA in the default directory you have to specify: VAXA::"TASK=EXAMPLE"

X.25 AND X.29 MODULES

X.25 AND X.29 MODULES

INTRODUCTION

The Packetnet System Interface (PSI) is designed to communicate with another system over a PPSN.
Using NCP you can display the different parameters that influence the function and performance of PSI.

In this module will be discussed what are the functions of the different software modules you can display with NCP.

OBJECTIVES

At the end of this module the student will understand the function of:

- the X.25 protocol module
- the X.25 server module
- the X.25 access module
- the X.29 server module

and understand how to display the parameters for these modules.

RESOURCES

VAX/VMS Networking Manual
VAX/VMS Network Control Program Reference Manual

PSI OVERVIEW

The Packetnet System Interface (PSI) is designed to follow the recommendations produced by the CCITT.

The major advantages of packet networks include cost savings for medium traffic, rates independent of distance, flexibility, network management provided by the value-added carries, and an ability to communicate with other vendor's equipment.

THE X.25 RECOMMENDATION

The end user does not need to know anything about the internal mechanism in Public Packet-Switched Networks (PPSN). Users may treat the PPSN as a magic cloud.

The X.25 recommendation specifies the protocols used to communicate between the local system and the network interface.

The local system is referred to as the Data Terminal Equipment (DTE).

There are two types of DTEs:

- Packet-Mode DTEs (generally a computer system)
- Start/Stop Mode DTEs (generally an asynchronous user terminal)

The network interface is called the Data Circuit Terminating Equipment (DCE).

The recommendation defines three levels (layers) of protocol which apply to the single physical link that connects the DTE to the network interface (DCE). They are:

- level 3, packet level
- level 2, link level
- level 1, physical level

Figure 10-1 shows how user data is made into a packet or packets, and how packets are made into frames.

Level 3 accepts data and adds a packet header to form a packet. The packet header contains the logical channel number, the identification of the packet type and may also have other control information.

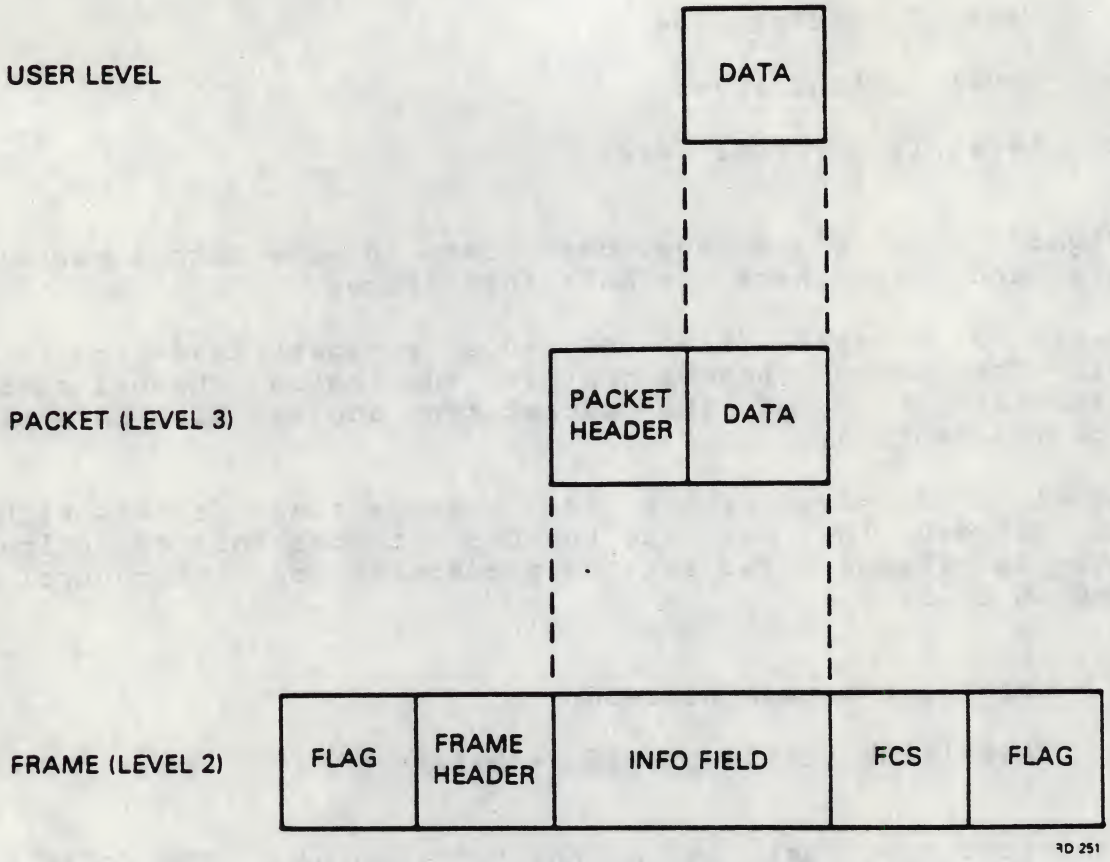
Level 2 is responsible for transferring packets without errors between the DTE and the DCE. It does this by enclosing packets in frames. Two sets of procedures for link control are defined in C.25:

- LAP (link access procedure)
- LAPB (link access procedure balanced)

The CCITT have based LAP on the HDLC protocol. The CCITT modified LAP to LAPB to make it compatible with a later version of HDLC. The DTE need only implement one set of procedures (LAP or LAPB).

Level 1 controls the physical transmission between the DTE and the DCE.

Figure 10-1: User Data, Packet, and Frame Format



Virtual Circuits

When you set up a call to a remote DTE, you are creating a link between the two DTEs. In a packet switching network, the physical link along which packets are sent is not specified. The link between the DTEs is a logical rather than a physical one. The logical link from a program on the local DTE to a program on a remote DTE is called a virtual circuit.

The packet switching software defines a number of channels, so that one physical line can be shared by more than one virtual circuit. The channels are called logical channels. Each channel is identified by a number called a logical channel number (LCN).

When a virtual circuit is created, the local DTE assigns one of the logical channels. Every packet which is sent over that virtual circuit will then contain the LCN of the logical channel used. The LCN is held as part of the packet header and is used to determine the address to which the packet is sent.

The local DCE identifies the virtual circuit from the logical channel, and determines which remote DCE the packet is sent to. The remote DCE assigns a logical channel to the remote DTE at the other end of the virtual circuit. The remote LCN and the local LCN are usually different.

Figure 10-2 shows a DTE which has three virtual circuits. One circuit uses separate logical channels to send data across the physical link to the DCE. There is a logical channel at each end of the virtual circuit, and these channels are independent of each other.

The virtual circuit between two DTEs may be temporary (switched), or permanent.

- Permanent Virtual Circuit (PVC)

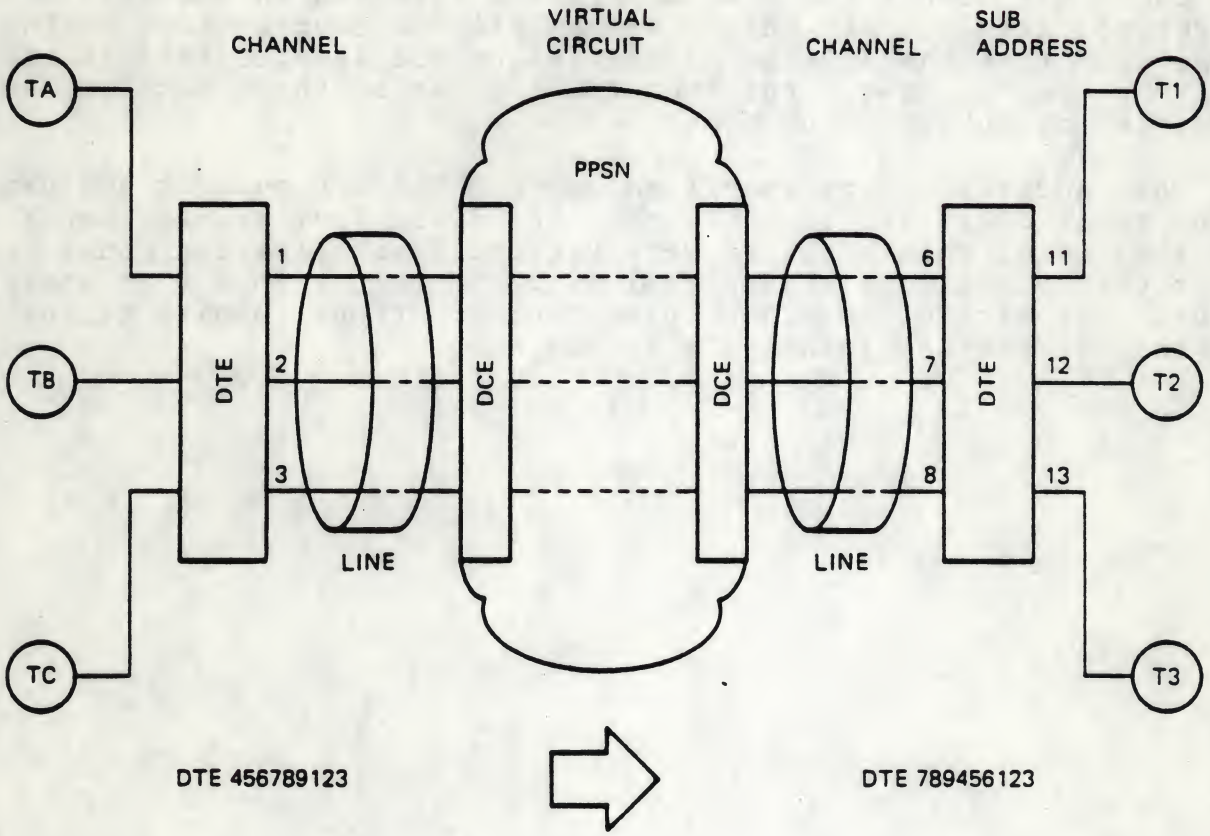
  If a circuit is permanent, it is similar to a leased line between the local DTE and the remote DTE. In a public network, logical channels are permanently assigned to the virtual circuit by the network authorities when the user subscribes to the network.

- Switched Virtual Circuit (SVC)

  A switched virtual circuit is similar to a dial-up line. The virtual circuit is created only when there is data to transmit, and it is cleared when the transfer is complete. The user at the local DTE sends a Call Request packet containing the address of a remote DTE. This causes the X.25 software to assign a logical channel and create a SVC.

The time taken to set up a call depends on whether the two DTEs which need to communicate are connected by a PVC or an SVC. A PVC connection means that there is no call setup time. SVCs are more flexible, allowing a DTE to communicate with any other DTE in the PSN.

Figure 10-2: Two-Node Configuration



Notes on Figure 10-2:

- TA calls T1 using remote subaddress 11

  TB calls T2 using remote subaddress 12

  TC calls T3 using remote subaddress 13

- T1, T2, and T3 respond using the incoming call identification data in the NCB

- Note that the channel numbers at each end of a virtual circuit are independent

DTE ADDRESSING

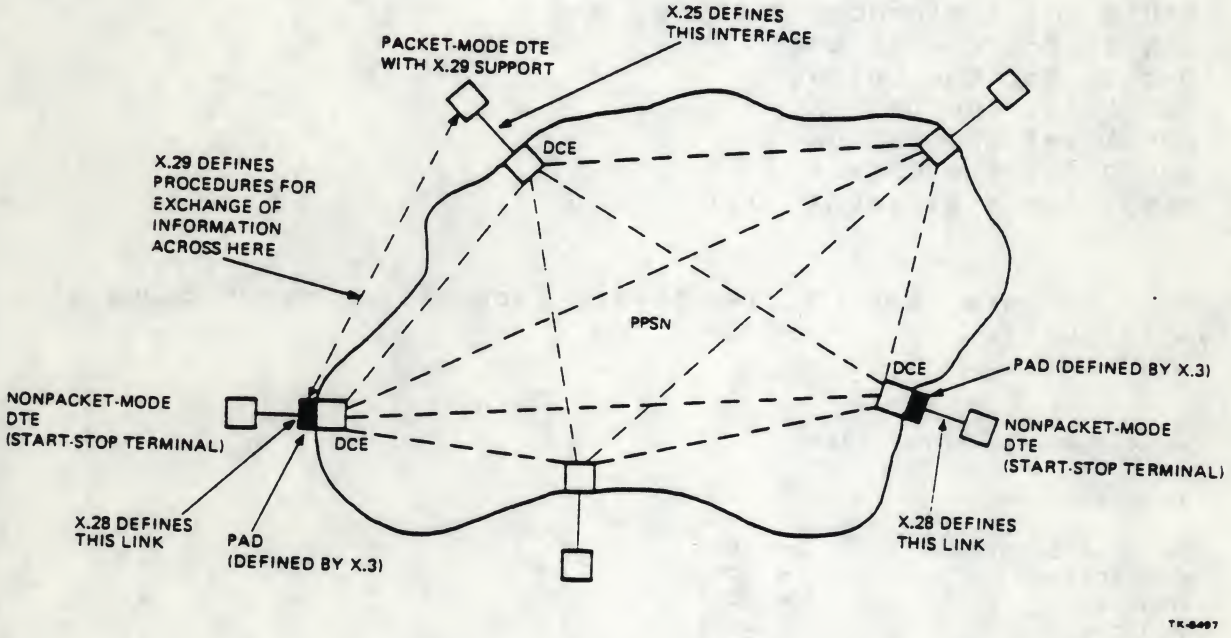
Each DTE connected to a PPSN is assigned an address. The addresses are totally unique across all PPSNs worldwide. During connection time the address of the remote DTE is specified in the Call Request packet. For PVCs the address of the remote DTE is specified at subscription time.

The addresses are specified as four-bit BCD numbers and can be up to 12 or 14 digits in length. The first four digits identify the PPSN, the so called Data Network Identification Code. Up to eight additional digits following the DNIC identify an individual DTE within the PPSN. Up to four additional digits following the DTE address identify a subaddress.

THE X.29 RECOMMENDATION

The X.3, X.28, and X.29 recommendations together specify how to control asynchronous terminals connected directly to the network. Together these protocols define the interactive Terminal Interface (ITI).

Figure 10-3: X.3, X.28, and X.29 Protocols



Notes on Figure 10-3:

X.3 Packet Assembly/Disassembly (PAD) facility in a public data network.

X.28 DTE/DCE interface for start-stop mode Data Terminal Equipment (DTE) accessing the Packet Assembly/Disassembly (PAD) facility in a public data network situated in the same country.

X.29 Procedures for the exchange of control information and user data between a packet-mode DTE and a PAD facility.

HARDWARE REQUIREMENTS FOR AN X.25 CONNECTION

When you want to connect to another system using X.25 you need a synchronous device, or synchronous port on a controller.

To get the names of the available controllers at the moment you need them, use to the Nfetworks and Communications Buyer's Guide.

The interfaces available according to the Buyer's Guide of 1988 July-September are:

KMV1A for a Microbus machine
DMB-32 for the BI-bus
DMB-32 for the Unibus
KMS11 for the Unibus
DUP11 for the Unibus
KMS1P for the Unibus
DST32 for a MicroVAX 2000

So, to use the X.25-software you need a synchronous line (Example 10-1).

```
NCP>SHOW LINE DUP-0 CHAR

Line = DUP-0

Receive buffers              = 14
Controller                   = normal
Duplex                       = full
Protocol                     = LAPB
Retransmit timer             = 2000
Maximum block                = 4101
Maximum retransmits          = 20
Maximum window               = 7
X.25 line interface mode     = DTE
Network                      = TRYIT
```

Example 10-1: A Synchronous Line Used for X.25

X.25 AND X.29 MODULES

There are three types of X.25:X.29 modules:

- X.25 Protocol Module
- X.25 Server and X.29 Server module
- X.25 Access module

X.25 PROTOCOL MODULE

The X.25 protocol module is the software component that controls the transmission of data packets over PSDNs. The database for the X.25 protocol module identifies the networks to which your DTEs are connected, defines the DTEs and specifies any user group to be associated with the DTEs.

Note that the term 'network name' in this context does not refer to a PSDN, but rather to a route to a PSDN.

Make sure that the profile-name is the correct one, if you want to connect to a public data network.

In the database the values of parameters that affect data-packet control. Defaults set the size and control the flow of data-packets over switched virtual circuits, and control setup and clearing of these circuits.

Your local DTE is identified by the DTE address and network name.

```
NCP> SHOW X25-PROTOCOL KNOWN DTE

Module X25-Protocol Volatile Summary as of 12-OCT-1988 14:21:45

  DTE             Network          State              Active     Active
                                                      Channels   Switched
  1234            TRYIT            on  -running          1          1
  3542            EDUNET           on  -running          1          1


NCP>SHOW X25-PROTOCOL KNOWN NETWORKS CHARACTERISTICS

Module X25-Protocol Volatile Characteristics as of 12-OCT-1988 14:24:13

Network                   = TRYIT
Profile                   = ISO8208

Network                   = SECOND
Profile                   = EDUNET
```

Example 10-2: Using NCP to Display the X.25 Protocol Characteristics

X.25 SERVER / X.29 SERVER MODULE

To handle calls coming in over a PSDN from remote DTES and terminals, the X.25 and X.29 server modules are needed. These modules are often referred to as call handlers. the X.25 server module handles incoming calls that originated at a remote DTE; the X.29 server module handles incoming calls that originated at a remote terminal.

The configuration database for the server modules defines the processes that are the destinations for calls, so that incoming calls from a PSDN can be directed to the appropriate destination. The server database specifies the maximum number of circuits the server module may have, that is: the maximum number of incoming and outgoing calls that all destinations can handle.

```
NCP>SHOW X25-SERVER CHARACTERISTICS

Module X25-Server Volatile Characteristics as of 12-OCT-1988 14:24:16

Maximum circuits                = 255

NCP>SHOW X29-SERVER CHARACTERISTICS

Module X29-Server Volatile Characteristics as of 12-OCT-1988 14:24:40

Maximum circuits                = 16
```

Example 10-3: X.25 and X.29 Server Characteristics

When you specify a subaddress in connecting a X.25-system, then the X.25-protocol module will check validity of the packets, and will direct the informationstream to the X.25-SERVER module.
There the subaddress field will be used to route you to the desired destination.

Look at example 10-4. If you are somewhere in the network, and you want to connect to the node NEWNOD, you first must specify the DTE number of the PSI-multihost node. The X.25-SERVER CHARACTERISTICS of this node are displayed in example 10-4.
If the DTE number of the multihostnode is 12345, you must specify a subaddress in the range 10-15. In this example we shall take 14. Then the X.25-SERVER at the multihost node will route you through (using DECnet) to the node NEWNOD.
At the node NEWNOD the software PSI-ACCESS must be installed to talk to PSI at the multihostnode with DTE number 12345.
The address you specify for node NEWNOD will be 1234514.

```
NCP> SHOW X25-SERVER KNOWN DESTINATIONS CHARACTERISTICS

Module X25-Server Volatile Characteristics as of 12-OCT-1988 14:24:59

Destination             = NODEX
Node                    = 40.1 (NODEX)
Object                  = 36
Priority                = 200
Subaddresses            = 6-10

Destination             = PSI_MAIL
Object                  = PSI_MAIL
Priority                = 100
Call mask               = FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Call value              = FF00000056332E30204D41494C2D3131

Destination             = NEWNOD
Node                    = 40.3 (NEWNOD)
Object                  = 36
Priority                = 200
Subaddresses            = 11-15

Destination             = X29_SERVER
Priority                = 127
Call mask               = FF
Call value              = 01
```

Example 10-4: The Destinations of the X.25 Server Module

X.25 ACCESS MODULE

The X.25 access module provides a means for user processes on VAX/VMS host nodes to access remote nodes or terminals connected to a PSDN through a VAX/VMS node serving as a multihost connector node. The host node must be configured with VAX PSI software in multihost mode. The X.25 access module identifies the connector node to which the local node is to be connected and the network the connector node can access.

```
NCP> SHOW X25-ACCESS KNOWN NETWORKS CHAR

Module X25-Server Volatile Characteristics as of 12-OCT-1988 14:26:43

Network                   = TRYIT
Node                      = 40.40 (NODEA)
```

Example 10-5: The X.25 Access Characteristics

USING THE X25-NETWORK

When you have the PSI-software installed, you can use it. The most common used commands are SET HOST/X29 and MAIL.

SET HOST/X29

To use SET HOST across an X25-network you must use the qualifier /X29. Then the host based PAD will be used. These has the parameter settings, specified by the CCITT specification X.3.

To set up a remote terminal session you have to specify the command SET HOST/X29.

To use the SET HOST/X29 command you first need to know what is the default datanetwork the PSI-software will choose.
For that reason you 'll have a look at the logical PSI$NETWORK.
If the system you want to connect to is not in the network specified by the logical PSI$NETWORK, you must prefix the DTE-number by the correct NETWORK-name.

In the next example the system you want to connect to is connected to the network EDUNET, DTE-number 3542.
If our node is connected to both networks, we can connect to it using the command SET HOST/X29 EDUNET%3542 (Example 10-6).

```
$ SHOW LOGICAL PSI$NETWORK
  "PSI$NETWORK" = "TRYIT"  (LNM$SYSTEM_TABLE)
$ SET HOST/X29 EDUNET%3542

WELCOME TO NODE ALLRIG

Username:
```

Example 10-6 Using SET HOST/X29 to an Other Network

MAIL

To use MAIL across the X25-network you must specify the DTE-number of the destination node before the number prefixed by the expression PSI%.

To address the user PETERS on the node with DTE-number 12345 the destination address will be PSI%12345::PETERS.

```
MAIL> SEND
To:   PSI%12345::PETERS
Subj: the answer is ....
      .....
      .....
```

Example 10-7 Sending a Mail Using PSI

```
MAIL>

You have 1 new message.

MAIL> READ 1                                                NEWMAIL

From:   PSI%TRYIT.<no-node>::SYSTEM
To:     PSI%12345::WILLIAMS
Subj:   New systems next month
        .......
        .......
```

Example 10-8 Receiving a Mail

DATALINK MAPPING

You can also use the datanetwork to use DECnet across it. Then all the normal DECnet traffic will use the X25-network.

Also Hello-messages will be sent across the network. Then all the normal DECnet functions can be used across the network, so the normal SET HOST and MAIL commands can be used.

```
NCP> SHOW KNOWN CIRCUITS

Known Circuit Volatile Characteristics as of 18-OCT-1988 13:23:44

  UNA-0              on                              40.18 (VAXQ)
  X25-DLM            on                              40.11 (MOUSE)


NCP>SHOW CIRCUIT X25-DLM CHARACTERISTICS

Circuit = X25-DLM

State                     = on
Cost                      = 10
Hello timer               = 15
Maximum recalls           = 100
Recall timer              = 60
Number                    = 30111
Owner                     = Executor
Usage                     = outgoing
Type                      = X.25
Verification              = disabled
Adjacent node             = 40.11 (MOUSE)
Listen timer              = 30
```

Example 10-9 A DLM-circuit to Node MOUSE

SUMMARY

To communicate over a PPSN we need some specific software. We need an X.25 Protocol module, which imnplements the X.25 protocol, an X.25 call handler: the X.25 Server module, and if necessary an X.25 Access module.

To setup a remote terminal session over a PPSN we need an X.29 server module.

With the NCP we can look at the characteristics and parameters needed to create a communication channel to another system over the PPSN.

To use the X.25-network we can us the SET HOST/X.29 command and the MAIL command.

DATABASES

DATABASES

INTRODUCTION

In this module is discussed how the network parameters are stored. For this purpose the network uses some databases.

In this module the different types of database are discussed, and the way to put information in the databases, to change information in the database and to read the information in the database.

OBJECTIVES

At the end of this module the student will understand which databases are used and how to manipulate to information in the databases.

RESOURCES

VAX/VMS Network Control Program Reference Manual
VAX/VMS Networking Manual

DATABASES

DECnet-VAX maintains databases to contain parameter information that the local node needs to participate in the network. There are three types of databases:

- Volatile database
- Permanent database
- PSI database

Volatile Database

The volatile database, maintained by NETACP, is a memory resident database containing current network configuration parameters. To manipulate and look in this database use the NCP SET, SHOW and CLEAR commands.

To read the volatile database we use the NCP SHOW command. To use this command we need no special privilege.

```
NCP> SHOW KNOWN OBJECTS

Known Object Volatile Summary as of  6-NOV-1988 12:02:18

   Object    Number  File/PID                      User Id          Password

  $MOM            0
  $NICONFIG       0
  SMISERVER       0  2020010C
  FAL            17  FAL.EXE
  HLD            18
  NML            19  NML.EXE
  REMACP         23  2020010F
  MIRROR         25
  EVL            26  2020010E
  MAIL           27  MAIL_SERVER.EXE
  PHONE          29  PHONE.EXE
  CTERM          42  2020010F
  DTR            63
  PCFS           64
```

Example 11-1: Using NCP to read the volatile database

To change or set a parameter in the volatile database we use the NCP SET command. For this command the OPER privilege is needed.

```
NCP> SET CIRCUIT UNA-0 COST 6
NCP> SHOW CIRCUIT UNA-0 CHAR

Circuit Volatile Characteristics as of 17-AUG-1987 09:45:26

Circuit = UNA-0

State                     = on
Service                   = enabled
Designated router         = 4.23 (VAXA)
Cost                      = 6
Router priority           = 64
Hello timer               = 15
Type                      = Ethernet
Adjacent node             = 4.17 (VAXE)
Listen timer              = 45
```

Example 11-2: Using NCP to modify the volatile database

There is a special case of the NCP SET command. The SET <comp> ALL command is used to update the volatile database with all the parameters of the <comp> stored in the permanent database. To use the NCP SET xxx ALL command you need SYSPRV.

To clear network parameters or reset them to the default values we use the NCP CLEAR command. To use the CLEAR command we need OPER privilege.

```
NCP> CLEAR CIRCUIT UNA-0 ALL
```

Permanent Database

The permanent database contains all the parameter information that the local node needs to function. The parameters set by the user or network manager are registered in the following files on disk:

NETCIRC.DAT
NETCONF.DAT
NETLINE.DAT
NETLOGING.DAT
NETNODE_LOCAL.DAT
NETOBJECT.DAT
NETX25.DAT
NETX29.DAT
NETNODE_REMOTE.DAT
NETPROXY.DAT

To set, change and read parameters in the permanent database we use the NCP DEFINE, PURGE and LIST command. For these commands the system privilege (SYSPRV) is required. In table 11-1 the files, containing the parameters, are listed.

Table 11-1: The Permanent Database Files

Filenaam	Contents
NETCIRC.DAT	Contains all the non-default values of the circuit parameters.
NETCONF.DAT	Contains the non-default parameter values for the ETHERNET module configurator.
NETLINE.DAT	Contains all the non-default values of the line parameters.
NETLOGING.DAT	Contains all the non-default parameters for the logging.
NETNODE_LOCAL.DAT	Contains all the non-default parameter values for the local host (executor).
NETOBJECT.DAT	Contains all the non-default parameters in the object database.
NETX25.DAT	Contains the non-default parameters of the X.25 server module and X.25 protocol module.
NETX29.DAT	Contains the non-default parameters of the X.29 server module.
NETNODE_REMOTE.DAT	Contains the local defined node characteristics of other nodes.
NETPROXY.DAT	Contains the defined proxy names. In VMS V4.* the name of the file was NETUAF.DAT.

PSI Database

Each PSI DTE that is connected to a PSDN has a database that supplies component and parameter information concerning the local DTE, local lines, virtual circuits, local modules, and local objects.

The VAX PSI configuration is also stored in the DECnet-VAX configuration. Just as with the DECnet-VAX configuration database, the VAX PSI configuration database consists of both a volatile and a permanent database.

SUMMARY

When the network is up and running, it stores several characteristics and counters in a database in memory: the volatile database. This database we can look at by using the NCP utility.

To get the initial parameter settings at startup time, the network looks at another database, which consists of several files on disk: the permanent database.

To get the information for setting up the network for accessing a PPSN the PSI database is used. This database consists of a volatile part and a permanent part on disk.

MONITORING

MONITORING

INTRODUCTION

To assist the network manager in achieving and maintaining good network performance, system managers of network nodes should closely monitor the network related aspects of their specific nodes.

There are several software tools to monitor the network, such as:

- Event Logging
- NCP
- MONITOR
- NMCC/DECnet Monitor
- VAX Ethernim

OBJECTIVES

At the end of this module the student will understand

- The use of the event logger
- The use of NCP to display counters
- The way the MONITOR command can be used for DECnet
- The function of NMCC/DECnet Monitor
- The function of VAX Ethernim

RESOURCES

VAX/VMS Networking Manual
VAX/VMS Network Control Program Reference Manual
VAX/VMS Monitor Utility Reference Manual

MONITORING

To assist the network manager in achieving and maintaining good network performance, system managers of network nodes should closely monitor the network related aspects of their specific nodes.

There are several software tools to monitor the network, such as:

- Event Logging
- NCP
- MONITOR
- NMCC/DECnet Monitor
- VAX Ethernim

EVENT LOGGING

DECnet software logs significant events that occur during network operation. An EVENT is a network on system-specific occurrence for which the logging component maintains a record. Significant events include:

- Circuit, line and node counter activity
- Changes in circuit, line and node states
- Lost event reporting
- Service requests
- Routing performance and error counters
- Data transmission performance and error counters

This information is useful for monitoring the network because it can be recorded continuously by the event logger (EVL).

You can control several aspects of event logging including:

- Source-related parameters
  - The actual events to be logged
  - The source component for which events are collected
  - The location where the events are logged
- Sink-related parameters
  - The name of the logging component at the local node
  - The operational state of the logging component at the local node

Events are defined by class and type (Table 12-1). For the most part, events are logged for the various DNA layers and for system-specific resources.

To display the logging status we use NCP:

NCP> SHOW KNOWN LOGGING

The result is shown in example 12-1.

```
NCP> SHOW KNOWN LOGGING

Known Logging Volatile Summary as of 17-AUG-1987 11:24:10

 Logging sink type = monitor

    Sink Node       Source                Events          State Name

   4.23 (VAXA)      (All sources)         0.0-9                on
                    (All sources)         2.0-1
                    (All sources)         4.2-13 15-16
                                          18-19
                    (All sources)         5.0-18
                    (All sources)         128.0-4
```

Example 12-1: The use of NCP to show the logging status

Table 12-1: Event Logging Classes

Class	Description
0	Network Management Layer
1	Applications Layer
2	Session Control Layer
3	End Communication layer
4	Routing Layer
5	Data Link Layer
6	Physical Link Layer
7 - 31	Reserved for Other Common Classes
32 - 63	RSTS System-Specific
64 - 95	RSX System-Specific
96 - 127	TOPS-20 System-Specific
128 - 159	VMS System-Specific
160 - 479	Reserved for Future Use

Sink-Related Parameters

Specifying the Logging Component

The logging component is the device or process that records the events released by the event logger (see Figure 12-1).

There are three types of logging components:

- Logging Console
- Logging File
- Logging Monitor

If no sink name is specified, defaults to Operator Communication (OPCOM) facility.!(VAX/VMS SPECIFIC).

Displays formatted event messages on all terminals enabled as NETWORK (using REPLY/ENABLE=NET), including the console (OPA0:)!(VAX/VMS SPECIFIC).

Specifying the Operational State

You can control the operational state of logging for the local node only. There are three logging states:

- HOLD
  - Indicates that the sink is temporarily unavailable
  - Events destined for that location are queued
- OFF
  - Indicates that the sink is unavailable for receiving event information
  - Events will not be logged for that sink
- ON
  - Indicates that the sink is available for receiving event information
  - This is the normal operational state

```
DECnet Event 4.7, Circuit down, circuit fault
From node 104 (BOSTON), Occurred 12-OCT-1986 12:37:14:24
Circuit UNA-0
line synchronization lost

DECnet Event 4.7, Circuit down, circuit fault
On node 8 (NASHUA), Occurred 28-SEPT-1986 19:57:09:31
Circuit DMC-0
Adjacent node listener receive timeout
```

Example 12-2: Event Logging Notices

NCP

We can use NCP as a monitor. Generally we can use it to monitor one of the four components:

- o node
- o line
- o circuit
- o object

NCP can display the following types:

- o counters
- o characteristics
- o status
- o summary
- o events

The COUNTER information provides us statistical information for nodes, lines and circuits.

The CHARACTERISTIC information includes the static information from the configuration database plus the parameters specified by the network manager.

The STATUS information includes dynamic information that usually reflects network operations for the running network.

The SUMMARY information includes only the most useful information from both static and dynamic sources.

The EVENTS information includes information about events currently being logged for the logging component (EVL). This display type is valid only for the commands SHOW LOGGING and LIST LOGGING.

For effective monitoring it is useful to write monitoring information to a logging file:

```
NCP> SHOW ACTIVE CIRCUIT SUMMARY TO SYS$MANAGER:NET.LIS
```

```
NCP> SHOW LINE UNA-0 COUNTERS

Line = UNA-0

       26853  Seconds since last zeroed
       38939  Data blocks received
       11323  Multicast blocks received
           0  Receive failure
     2927668  Bytes received
     1408038  Multicast bytes received
           0  Data overrun
           0  Local buffer errors
       33104  Data blocks sent
        2414  Multicast blocks sent
           0  Blocks sent, multiple collision
           0  Blocks sent, single collision
          61  Blocks sent, initially deferred
     2092951  Bytes sent
      250074  Multicast bytes sent
           0  Send failure
           0  Collision detect check failure
          20  Unrecognized frame destination
           0  System buffer unavailable
           0  User buffer unavailable
```

Example 12-3: The use of NCP to display the line counters

```
NCP> SHOW CIRCUIT UNA-0 COUNTERS

Circuit = UNA-0

       26485  Seconds since last zeroed
       11353  Terminating packets received
       11636  Originating packets sent
           0  Terminating congestion loss
           0  Transit packets received
           0  Transit packets sent
           0  Transit congestion loss
           0  Circuit down
           0  Initialization failure
       13561  Data blocks sent
     1164399  Bytes sent
       22560  Data blocks received
     2491338  Bytes received
           0  Unrecognized frame destination
           0  User buffer unavailable
```

Example 12-4: The use of NCP to display the circuit counters

```
NCP> SHO EXECUTOR COUNTERS

Node Counters as of 17-AUG-1987 15:23:53

Executor node = 4.23 (VAXA)

       25416  Seconds since last zeroed
      173446  Bytes received
      174861  Bytes sent
        3868  Messages received
        3887  Messages sent
          16  Connects received
          19  Connects sent
           8  Response timeouts
           0  Received connect resource errors
           6  Maximum logical links active
           0  Aged packet loss
           0  Node unreachable packet loss
           0  Node out-of-range packet loss
           0  Oversized packet loss
           0  Packet format error
           0  Partial routing update loss
           0  Verification reject
```

Example 12-5: The use of NCP to display the executor counters

```
NCP> SHO NODE VAX COUNTERS

Node Counters as of 17-AUG-1987 15:43:48

Remote node =   4.13 (VAX)

       18255  Seconds since last zeroed
       69152  Bytes received
         522  Bytes sent
         150  Messages received
         142  Messages sent
           0  Connects received
           5  Connects sent
           2  Response timeouts
           0  Received connect resource errors
```

Example 12-6: The use of NCP to display node counters

MONITOR

We can use the MONITOR utility to display the actual use of buffers by DECnet.

```
$ MONITOR DECNET

                    VAX/VMS Monitor Utility
                    DECNET STATISTICS
                    on node NOOT
                    17-AUG-1987 11:46:57

                                  CUR       AVE       MIN       MAX
Arriving Local Packet Rate       0.00      1.10      0.00      1.66
Departng Local Packet Rate       0.00      1.10      0.00      1.66

Arriving Trans Packet Rate       0.00      0.00      0.00      0.00
Trans Congestion Loss Rate       0.00      0.00      0.00      0.00

Receiver Buff Failure Rate       0.00      0.00      0.00      0.00
LRPs Available                  48.00     48.66     48.00     49.00
```

Example 12-7: The use of the VAX Monitor Utility

NMCC/DECNET MONITOR

The Network Management Control Center/DECnet Monitor is a set of sophisticated tools for the observation and control of complex networks. You can command the tools to present color graphics displays that show the condition of the network. The tools also work with a database of configuration information and network parameters; you can access and analyze the information and tune the network. The NMCC/DECnet Monitor is a layered product that runs on most VAX/VMS systems.

VAX ETHERNIM (ETHERNET NETWORK INTEGRITY MONITOR)

VAX ETHERnim is a network maintenance application program for Ethernet. It tests the communications path to nodes on the Ethernet. It also maintains a database containing information about each node.

SUMMARY

To monitor the network we can use the EVENT LOGGER. Then the events can be logged in a file, at a console or in a file to use as input for a monitor utility.

The events in the network are identified by an event type number.

The second way to monitor is by using NCP. The NCP SHOW command for the volatile database, and the NCP LIST for the permanent database.

Another way to monitor the network use is by the MONITOR utility.

APPENDIX A: NCP

APPENDIX A: NCP

NCP

The network Control Program (NCP) is a DECnet-VAX utility that accepts terminal commands to configure, control, monitor, and test a DECnet network.

INVOKING AND EXITING THE NCP-UTILITY

To invoke NCP, issue the DCL command.

```
$RUN SYS$SYSTEM:NCP
```

NCP will return the following prompt:

```
NCP>
```

Once you receive this prompt, you can issue NCP commands. To exit NCP, type EXIT or press CTRL/Z After the NCP> prompt.

Alternatively, you can execute a single NCP command by using a DCL string assignment statement. For example:

```
$ NCP:==$NCP
$ NCP SHOW STATUS KNOWN LINES
```

Directing Output

Output for the SHOW and LIST commands is normally displayed on the default output device, SYS$OUTPUT. Alternatively, you may direct output to a specified file using the TO qualifier with the SHOW or LIST command.

Privileges/Restrictions

Certain privileges are required to perform most NCP commands. The only commands that can be performed without privileges are CLEAR EXECUTOR NODE, HELP, LOOP EXECUTOR, LOOP LINE, LOOP NODE, SET EXECUTOR NODE, SHOW, and TELL.

The PURGE, DEFINE, LIST and SET xxx ALL commands require the system privilege (SYSPRV) or some other way to access files with system protection, and the remaining NCP commands require the operator privilege (OPER). Note that at any time you may use access control parameters with the SET EXECUTOR NODE command to activate the network management listener (NML) to run under an account with the OPER privilege.

You must have the DETACH, CMKRNL, SYSNAM, and SYSPRV privileges to start the network when it is not currently active.

Commands

Syntax

```
NCP> command component-name parameter [...][qualifier][...]
```

NCP commands

```
CLEAR/PURGE CIRCUIT
CLEAR/PURGE EXECUTOR
CLEAR EXECUTOR NODE
CLEAR/PURGE LINE
CLEAR/PURGE LOGGING
CLEAR/PURGE LOGGING EVENTS
CLEAR/PURGE LOGGING NAME
CLEAR/PURGE MODULE X25-ACCESS
CLEAR/PURGE MODULE X25-PROTOCOL
CLEAR/PURGE MODULE X25-SERVER/X29-SERVER
CLEAR/PURGE NODE
CLEAR/PURGE NODE CIRCUIT
CLEAR/PURGE OBJECT
CONNECT NODE
CONNECT VIA
COPY KNOWN NODES
DISCONNECT LINK
HELP
LOAD NODE
LOAD VIA
LOOP CIRCUIT
LOOP EXECUTOR
LOOP LINE
LOOP NODE
PURGE MODULE CONFIGURATOR
SET CIRCUIT ALL
SET/DEFINE CIRCUIT
SET/EXECUTOR ALL
SET/DEFINE EXECUTOR
SET EXECUTOR NODE
SET LINE ALL
SET/DEFINE LINE
SET LOGGING ALL
SET/DEFINE LOGGING EVENTS
SET/DEFINE LOGGING STATE
SET/DEFINE MODULE CONFIGURATOR
SET/DEFINE MODULE X25-ACCESS
SET/DEFINE MODULE X25-PROTOCOL
SET/DEFINE MODULE X25-SERVER/X29-SERVER
SET NODE ALL
```

```
SET/DEFINE NODE
SET/DEFINE NODE CIRCUIT
SET OBJECT ALL
SET/DEFINE OBJECT
SHOW AREA
SHOW/LIST CIRCUIT
SHOW/LIST EXECUTOR
SHOW/LIST LINE
SHOW LINKS
SHOW/LIST LOGGING
SHOW/LIST MODULE CONFIGURATOR
SHOW/LIST MODULE X25-ACCESS
SHOW/LIST MODULE X25-PROTOCOL
SHOW/LIST MODULE X25-SERVER/X29-SERVER
SHOW/LIST NODE
SHOW/LIST OBJECT
TELL
TRIGGER NODE
TRIGGER VIA
ZERO CIRCUITS
ZERO EXECUTOR
ZERO LINE
ZERO MODULE X25-PROTOCOL
ZERO MODULE X25-SERVER/X29-SERVER
ZERO NODE
```